UTILISATIONS DE L'ORDINATEUR

MODELES D'EXPRESSION GRAPHIQUE

jean-pierre blanger

P.S.I.

UTILISATIONS DE L'ORDINATEUR

MODELES D'EXPRESSION GRAPHIQUE

La collection « **UTILISATIONS DE L'ORDINATEUR** »

Cette collection s'intéresse aux utilisations de l'ordinateur par domaine, chaque livre présente l'analyse de celui-ci et des programmes en Basic le plus souvent.

Sept titres sont actuellement disponibles dans cette collection ; certains disposent d'une disquette sur Apple II*.

— * Le Basic et l'école — tomes 1 et 2 — Jacques Gouet
— * Nouvelle comptabilité sur Apple II — tomes 1 et 2 — Gérard Llilio et Serge Llilio
— * La paie et ses annexes sur Apple II — Jean-Michel Jégo
— Les finances familiales — Jean-Claude Barbance (ISBN 2-86595-059-X)
— Outils financiers et comptables pour l'entreprise — Bernard Sulmon
— Tests statistiques usuels — Association Aemavi
— * Modèles d'expression graphique — Jean-Pierre Blanger

RAPPELS

Les séries :

En fait, il faudrait parler de niveaux, puisque la couleur attachée à chaque ouvrage permet de situer la « force » de celui-ci selon le code suivant :

Série VERTE : ouvrage d'initiation ne nécessitant que des connaissances de base.
Série BLEUE : suppose une connaissance élémentaire du sujet traité.
Série ROUGE : ouvrage d'approfondissement, niveau de complexité moyen.
Série NOIRE : ouvrage d'approfondissement, niveau de complexité élevé.

Les collections :

Les ouvrages d'Edition du PSI, actuellement au nombre de 100, sont répartis en collections : « **LANGAGES** », « **MATERIELS** », « **PROGRAMMES** », « **GUIDES PRATIQUES** », « **MEMENTOS** », « **UTILISATIONS DE L'ORDINATEUR** », « **LOGIGUIDE** » et pour l'initiation, outre quelques livres hors collection, « **... POUR TOUS** ».

La loi du 11 mars 1957 n'autorisant, aux termes des alinéas 2 et 3 de l'article 41, d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective » et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale, ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause, est illicite » (alinéa 1er de l'article 40).

Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait donc une contrefaçon sanctionnée par les articles 425 et suivants du Code Pénal.

ISBN : 2-86595-122-7

c **Editions du P.S.I., B.P. 86, 77402 Lagny-sur-Marne (France)**

c Editions du P.S.I., B.P. 86, 77402 Lagny-sur-Marne (France)

1984
Imprimé en France

jean-pierre blanger

MODELES D'EXPRESSION GRAPHIQUE

Editions du P.S.I.
1984

Présentation de l'auteur

Jean-Pierre BLANGER *est élève ingénieur à l'ESIEA, Ecole Supérieure d'Informatique-Electronique-Automatique.*

Passionné d'informatique depuis 1978, il s'intéresse vivement aux micro-ordinateurs et à leurs applications.

Auteur d'articles et d'ouvrages, organisateur de réunions d'information, il aime partager sa passion avec les personnes dont le principal trait de caractère est la curiosité.

Du même auteur aux Editions du P.S.I.

MODELES PRATIQUES DE DECISION Tome 1
MODELES PRATIQUES DE DECISION Tome 2

COLLECTION PROGRAMMES

Sommaire

PRESENTATION 7

I EXPRESSION GRAPHIQUE PROGRAMMEE

- TRACES GRAPHIQUES DE BASE

PET/CBM - APPLE - TRS-80 10
Point 15
Vecteur 23
Polygones 30
Polygone régulier 36
Ellipse 42

- VARIATION SUR LES TRACES GRAPHIQUES DE BASE

Points aléatoires 48
Vecteurs aléatoires 53
Vecteurs paramétrés 59
Polygones aléatoires 68

II EXPRESSION GRAPHIQUE MATHEMATIQUE

- REPRESENTATION GRAPHIQUE D'EQUATIONS

Hypo et Epi-cycloïdes 74
Fonctions circulaires 84

III EXPRESSION GRAPHIQUE PAR L'OUTIL

- TRANSFORMATIONS GRAPHIQUES

Translation d'un point 100
Rotation d'un point 107
Translation d'un polygone 117
Rotation d'un polygone 124

- SCENES EN DEUX DIMENSIONS

Définition de surfaces 134
Translation de surfaces 142
Rotation de surfaces 149
Agrandissement de surfaces 156
Hachurage de surfaces 164
Elimination des parties cachées 180

- COURBES PLANES

Bezier 196
B-Spline 205

IV EXPRESSION GRAPHIQUE INTERACTIVE

Tracés interactifs 216

ANNEXE - GESTION GRAPHIQUE DE L'ECRAN 224

DISQUETTE D'ACCOMPAGNEMENT

Pour Apple II, IIplus, et *IIe*

Dos 3.3 version 48K ou plus

Vous avez acheté le livre "**MODELES D'EXPRESSION GRAPHIQUE**" de Jean-Pierre Blanger et vous n'avez pas le temps nécessaire pour entrer vous-même les programmes sur votre AppleII. Dans ce cas, la disquette que vous proposent les Editions du P.S.I. vous est tout particulièrement destinée.

CETTE DISQUETTE DOIT ETRE CONSIDEREE COMME UNE AIDE AU LECTEUR ET NON COMME UN PROGICIEL GRAPHIQUE.

L'ensemble des programmes présentés constitue la réplique exacte des listings proposés dans l'ouvrage.

Disponible en MARS 1984

Présentation

MODELES D'EXPRESSION GRAPHIQUE veut répondre à un double objectif : proposer aux débutants une approche progressive et didactique des possibilités graphiques d'un ordinateur individuel et à l'amateur éclairé des méthodes de résolution de problèmes complexes ainsi que des schémas types de structures de programmes évolués de dessin.

Pour atteindre ce double but, cet ouvrage ne pouvait présenter la réalisation d'un progiciel de dessin assisté par ordinateur. A l'opposé, il présente un ensemble de modèles qui traitent du plus simple au plus compliqué. Ainsi, au fil des pages, le lecteur est aussi bien mis face au problème du tracé d'un point ou d'une surface sur l'écran que face au problème de l'élimination de parties cachées dans une scène à deux dimensions ou du tracé de courbes planes.

Les modèles d'expression graphique sont proposés suivant les approches suivantes :

. **EXPRESSION GRAPHIQUE PROGRAMMEE :** C'est la première partie du livre. Elle présente une approche du graphisme sur ordinateur par des modèles grâce auxquels tout dessin réalisé est immuable. Ainsi, un dessin achevé sera synonyme de programme terminé. Après une initiation avec les tracés graphiques de base, on peut le constater grâce aux variations sur les tracés graphiques de base.

. **EXPRESSION GRAPHIQUE MATHEMATIQUE :** La seconde partie de l'ouvrage aborde le domaine de la représentation graphique d'équations. Sans traiter ce vaste sujet dans sa totalité, elle appréhende les fonctions paramétrées par un angle et montre

que si, cette fois encore, le programme est figé, il n'est pas immuable. En effet, le changement de paramètres modifie le dessin et on ne peut plus parler d'expression graphique programmée.

. **EXPRESSION GRAPHIQUE PAR L'OUTIL** : Cette partie est la plus longuement traitée de l'ouvrage car c'est la plus importante. Elle aborde les transformations graphiques de façon simple pour les employer ensuite dans la mise en forme de scènes en deux dimensions. Enfin, elle traite le problème des courbes planes. Cette démarche permet de résoudre quelques problèmes cruciaux du dessin sur ordinateur individuel et montre comment s'affranchir d'un programme pour réaliser des représentations graphiques.

. **EXPRESSION GRAPHIQUE INTERACTIVE** : La dernière partie de l'ouvrage présente l'ébauche d'un système intéractif que le lecteur à la charge d'améliorer. En utilisant les modèles proposés dans les approches précédentes, il ne doit avoir aucun mal à créer son propre système interactif digne de ceux actuellement réalisés pour la CAO ou le DAO ... qui sait ?

L'ensemble des modèles proposés est l'occasion d'un ou plusieurs programmes. Ceux-ci sont commentés et accompagnés d'un exemple d'exécution visant à une assimilation rapide et efficace de chacun de leurs effets.

Première partie

EXPRESSION GRAPHIQUE PROGRAMMÉE

Tracés graphiques de base

PET / CBM - APPLE - TRS 80

Avant de réaliser de nombreuses figures complexes, il faut se donner les moyens de faire des choses simples. Aussi, avant de tracer une courbe, il faut savoir tracer une droite et, avant de tracer une droite, il faut savoir tracer un point. Car, si l'on s'en réfère à la théorie de la géométrie Euclidienne, que tout jeune français appréhende dès son plus jeune âge, une droite est constituée d'une infinité de points.

Comment allumer un point sur un écran vide ? Voici le problème sur lequel nous allons tenter de faire le...

A l'origine, nous disposons d'un micro-ordinateur et, suivant son modèle, différents moyens sont disponibles.
Ainsi, sur :

L E P E T / C B M

Pour allumer un point, la seule possibilité offerte par le constructeur consiste à utiliser l'instruction **PRINT.** Celle-ci est pourvue des caractères de contrôle suivants :

- ♥ Effacement écran
- S Position home
- Q Passe à la ligne suivante
- O Passe à la ligne précédente
- J Décale d'une colonne vers la droite
- I Décale d'une colonne vers la gauche
- R Mise en vidéo inversée
- _ Retour en vidéo normale

qui, utilisés comme des caractères normaux, permettent l'édition

dans une position quelconque de l'écran, d'un point ou plutôt d'un caractère.

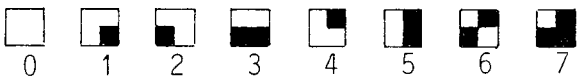
La résolution du PET/CEM reste très basse, c'est-à-dire qu'il n'est possible d'allumer que très peu de points sur la largeur et sur la hauteur de l'écran. Elle n'est, en effet, que de 40 x 24 points, autant qu'il y a de caractères par ligne.

Les possibilités graphiques offertes par cette machine sont alors très réduites car la précision des tracés est trop médiocre. Afin de pallier cette carence graphique, le PET/CBM est pourvu d'un ensemble très complet de caractères graphiques préprogrammés accessibles sous le mode de fonctionnement **'Majuscule'**

- **Mode Majuscule : POKE 59468,12**
- **Mode Minuscule : POKE 59468,14**

Ces caractères permettent de réaliser des représentations graphiques très réalistes, mais ils sont d'un usage très délicat et même quasi-impossible pour satisfaire les besoins que nous allons révéler au cours de nos réalisations.

Cependant, ô miracle, ces caractères graphiques préprogrammés permettent aussi d'augmenter la résolution du PET/CBM jusqu'à 4000 points (80 x 50). C'est par un arrangement ingénieux de seize caractères que la routine **'gestion d'écran'** permet de doubler les performances graphiques de la machine.



Les caractères ci-dessus permettent d'augmenter la précision des tracés par un facteur 2, car, quelque soit le cas de figure, il est possible d'utiliser les caractères 1, 2, 4, 8 pour représenter un point, les caractères 3, 5, 6, 9, 10, 12 pour représenter deux points, les caractères 7, 11, 13, 14 pour représenter trois points, le caractère 15 pour représenter quatre points et enfin le caractère 0, pour ne représenter aucun point.

Alors, même si la définition d'un tracé reste imprécise, la mise en oeuvre de représentations graphiques devient néanmoins possible. (Voir annexe : GESTION GRAPHIQUE DE L'ECRAN DU PET/CBM)

L ' A P P L E

Pour allumer un point, plusieurs possibilités sont offertes :

- **La première consiste à utiliser l'instruction PRINT qui, en conjonction avec les instructions HTAB et VTAB permet d'allu-**

mer un point sur l'écran quelle qu'en soit la position souhaitée.

En effet, il suffit, après avoir placé le curseur en position **HOME**, de donner le nombre de colonnes à passer à l'instruction **HTAB** ainsi que le nombre de lignes à passer à l'instruction **VTAB** pour atteindre une position donnée de l'écran. Notons que cette position peut être connue facilement puisque les octets 36 et 37 contiennent, à une unité près, les numéros de colonne et de ligne où se trouve le curseur.

Dans cette première façon d'opérer, la résolution (40 x 24) de l'Apple est trop basse pour réaliser des représentations graphiques réalistes mais cela n'a pas d'importance, car nous ne sommes pas au bout des ressources graphiques de cette machine.

. La seconde possibilité consiste à utiliser l'APPLE en mode graphique 'basse résolution'.

Pour entrer dans ce mode, il suffit d'utiliser l'instruction **GR.** L'écran se trouve alors divisé en deux zones. La première, en bas de l'écran est constituée de quatre lignes destinées à l'édition de texte à l'aide de l'instruction **PRINT.** La seconde zone occupe toute la partie haute de l'écran et permet une utilisation graphique sur la base d'une résolution 40 x 40 ainsi que l'utilisation de 16 couleurs.

La sélection des couleurs est effectuée par une ou plusieurs instructions **COLOR** = C ou :

C = 0	Noir
C = 1	Magenta
C = 2	Bleu foncé
C = 3	Violet
C = 4	Vert foncé
C = 5	Gris 1
C = 6	Bleu
C = 7	Bleu clair
C = 8	Marron
C = 9	Orange
C = 10	Gris 2
C = 11	Rose
C = 12	Vert
C = 13	Jaune
C = 14	Bleu turquoise
C = 15	Blanc

Les couleurs ayant été spécifiées, l'utilisation de l'instruction **PLOT X, Y** permet d'allumer un point à la **colonne X** et en **ligne Y.** Les instructions **HLIN** et **VLIN**, quant à elles permettent le tracé de droites Horizontales et Verticales.

Notons que l'utilisation de couleurs est extrêmement intéressante car, à tout moment, un point peut être effacé avec C = 0 et qu'il est possible d'agrandir la zone graphique afin qu'elle remplisse l'écran par une commande POKE - 16302,0 ce qui amène la résolution

jusqu'à 40 x 48 points.

. **La troisième et dernière possibilité offerte par l'APPLE est le mode 'Haute résolution'.**

Comme dans le mode précédent, il est possible de conserver quatre lignes de texte en tapant **HGR**, ce qui mène la résolution à 280 x 160 points, ou encore, en tapant **HGR2**, on peut obtenir une résolution maximum de 280 x 192 points.

Dans ce mode, bien que les couleurs soient soumises à des contraintes et que leur variété soit réduite pour des raisons technologiques, la précision des tracés devient extrêmement fine.

L'instruction PLOT du mode basse résolution devient **HPLOT** et remplace HLIN et VLIN. Ainsi, pour tracer un segment de droite, on écrira :

```
HPLOT XØ,YØ TO X1,Y1
```

pour tracer une ligne brisée :

```
HPLOT XØ, YØ TO X1,Y1 TO X2,Y2
```

Quant à la sélection des couleurs, elle se fait maintenant par l'instruction **HCOLOR = C** ou :

```
C = Ø   Noir 1
C = 1   Violet pour les colonnes impaires
C = 2   Vert pour les colonnes paires
C = 3   Blanc 1
C = 4   Noir 2
C = 5   Bleu pour les colonnes impaires
C = 6   Marron pour les colonnes paires
C = 7   Blanc 2
```

Une instruction supplémentaire, **TEXT**, est disponible en mode haute résolution et elle permet un retour des quatre dernières lignes de l'écran vers l'édition de texte sur la totalité de la vidéo. Ce mode graphique est sans aucun doute le plus satisfaisant étant donné la précision qu'il apporte et parce qu'on peut l'utiliser en plein écran par un POKE - 16302,0.

Au cours de ce livre, nous utiliserons ce mode graphique pour les exemples d'exécution, même si sur APPLE IIe on dispose maintenant d'une résolution monochromatique de 560 x 192 points grâce à la carte couleur "chat mauve" car ce n'est pas le cas de toutes les machines de ce type.

L E T R S 8 0

Pour allumer un point, deux possibilités sont offertes :

. **La première utilise les instructions SET et RESET.** On considère dans ce cas que l'écran est divisé en 48 lignes de 128 positions ce qui donne une résolution de 6144 points.

A une position donnée de l'écran deux opérations sont possibles. En supposant que l'on regarde le point situé à la **ligne X** et en **colonne Y** on peut l'allumer en tapant **SET(Y,X)** puis l'éteindre en tapant **RESET(Y,X)**.

Les représentations graphiques réalisées par ce procédé sont très imprécises, étant donnée la définition même du quadrillage de l'écran car les points, une fois allumés sont plus hauts que larges.

. **La seconde possibilité consiste à considérer que l'écran est divisé en 16 lignes de 64 points** ce qui fait un total de 1024 positions dont chacune peut recevoir un caractère graphique préprogrammé. Celui-ci étant un assemblage de 6 points du mode précédent, on établit que 1024 x 6 = 6144 et l'on trouve une relation entre les deux modes graphiques.

On comprend alors pourquoi l'ensemble des caractères graphiques préprogrammés représente l'ensemble des combinaisons de points sur six positions.

Ces caractères sont édités par une extension de l'instruction PRINT, soit **PRINT@X,CHR$(Y)** où **X** est le numéro de position et **Y** le caractère.

Notons encore l'existence de l'instruction **PRINT@X,STRINGS(Y,Z)** qui édite **Z** fois de suite le caractère dont le code est **Y** à partir de la position **X** et que le changement de mode est effectué par **PRINT CHR$(22)**.

CONCLUSION

Ces trois ordinateurs individuels comptent parmi les plus célèbres et montrent parfaitement qu'il n'existe pas de standard en ce qui concerne les possibilités graphiques. Alors, écrire des programmes compatibles avec toutes les machines est impossible. Cela est vrai mais, il est possible de limiter l'incompatibilité. C'est ce qui est fait dans ce livre. Pour atteindre ce but, les traitements sont réalisés avec un Basic standard et minimum que l'on trouve sur l'ensemble des P.S.I. Les affichages graphiques utilisent des sous-programmes spécialisés qui sont les seules parties des programmes à modifier d'une machine à l'autre. Les instructions graphiques utilisées sont celles de l'APPLE (il fallait faire un choix) mais, celles qui permettent d'allumer des points sont les seules mises en oeuvre, car toutes les machines n'ont pas la possibilité d'effacer ou d'éteindre des points.

Les options de programmation prises ici, doivent faciliter dans une large mesure l'adaptation des programmes et éliminer l'obstacle du choix du matériel cible à l'avantage du choix des qualités graphiques d'une machine.

POINT

Nous allons, dans ce chapitre, réaliser une routine permettant d'allumer un point sur l'écran. C'est au niveau de ce sous-programme qu'interviendront les modifications nécessaires à la transposition, sur différentes machines, de tous les programmes proposés dans ce livre. Pour les PET/CBM, par exemple, cette routine constitue l'interface entre le langage machine de GESTION GRAPHIQUE DE L'ECRAN (annexe) et le langage BASIC des programmes.

HYPOTHESE

Jusqu'à présent, nous avons vu la capacité et les facilités graphiques offertes par différents micro-ordinateurs. Nous en avons conclu que nous allions créer un sous-programme spécialisé pour allumer un point. Ce sous-programme c'est ***POINT***. Nous allons le mettre en oeuvre sur APPLE et PET/CBM afin de montrer un exemple d'adaptation qui peut paraître complexe au départ et qui, pourtant reste très simple lorsque l'on est méthodique.

MODELE

Avant de voir comment nous allons écrire le sous-programme ***POINT*** pour le PET/CBM, notons que pour d'autres machines (l'APPLE, par exemple), il suffit souvent d'une seule instruction (HPLOT) pour remplacer cette routine. Dans un tel cas, il est cependant préférable de conserver le sous-programme ***POINT*** que nous allons écrire, car il permet, par la suite, de n'avoir qu'à taper les lignes supplémentaires des programmes proposés dans les chapitres suivants au lieu d'avoir de nombreuses modifications à apporter. En outre, cela permet de développer des programmes facilement adaptables à n'importe quelle machine et c'est notre but.

Dans le cas de l'APPLE, il faut connaître l'effet de l'instruction **HGR** suivi de **POKE-16302,Ø**, qui **passe l'écran en mode graphique plein écran** et l'effet de l'instruction **HPLOT X,Y** qui **allume le point des coordonnées X et Y** tel que nous le verrons lors de l'exemple d'exécution. Ces trois commandes BASIC suffisent à l'écriture du sous-programme ***POINT***. Ce dernier, cependant, est aussi l'occasion de tester la validité des coordonnées du

point à afficher. C'est pourquoi nous testons les valeurs des variables X et Y par rapport aux valeurs minimum et maximum des axes de l'écran qui sont de 0 à 279 suivant X et 0 à 191 suivant Y. On retrouve bien la résolution de 280 x 192 points de l'APPLE.

Dans le cas du PET/CBM, le sous-programme ***POINT*** effectue aussi deux opérations. La première est de tester les domaines de validité des variables X et Y après avoir calculé les arrondis de leurs valeurs. La seconde est d'assurer l'interface entre le programme BASIC et le niveau langage machine afin de permettre l'affichage, c'est-à-dire l'allumage, du point des coordonnées X, Y. Alors, lorsque la valeur arrondie de X est comprise entre Ø inclus et 50 exclus et la valeur arrondie de Y est comprise entre Ø inclus et 80 exclus, les variables X et Y sont 'POKEes' — transmises au programme ***GESTION GRAPHIQUE DE L'ECRAN*** — respectivement aux adresses 975 et 976 où le sous-programme de gestion d'écran est censé trouver les coordonnées du point à allumer lorsqu'il est activé par l'instruction SYS826.

EXEMPLE D'UTILISATION

Les programmes exemples utilisent les sous-programmes ***TRACE*** qui permettent l'introduction successive de plusieurs couples de coordonnées pour se familiariser avec l'écran de chaque matériel.

Lors de l'exécution, si nous entrons les couples suivants :

PET/CBM

X	Y
5	5
6	5
5	6
7	7
8	8
9	9
10	10
11	11
11	12
-1	-1

APPLE

X	Y
10	10
20	20
40	40
80	80
85	80
80	85
85	85
90	90
91	91
-1	-1

nous obtenons les résultats en haut de la page 22.

Ou encore, en introduisant successivement les couples suivants :

PET/CBM

X	Y
15	15
35	0
55	15
35	35
35	15
0	35
79	35
0	0
79	0
-1	-1

APPLE

X	Y
0	0
279	0
0	159
279	159
150	0
150	159
0	75
279	75
150	75
-1	-1

nous obtenons les résultats en bas de la page 22.vante.

LE PROGRAMME

Traitement

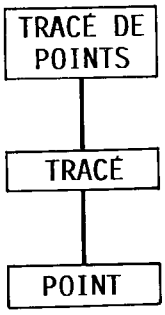
.Programme principal

Ensemble de couples de coordonnées → TRACÉ DE POINTS → *Affichage de points*

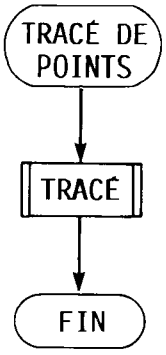
.Sous- programme POINT

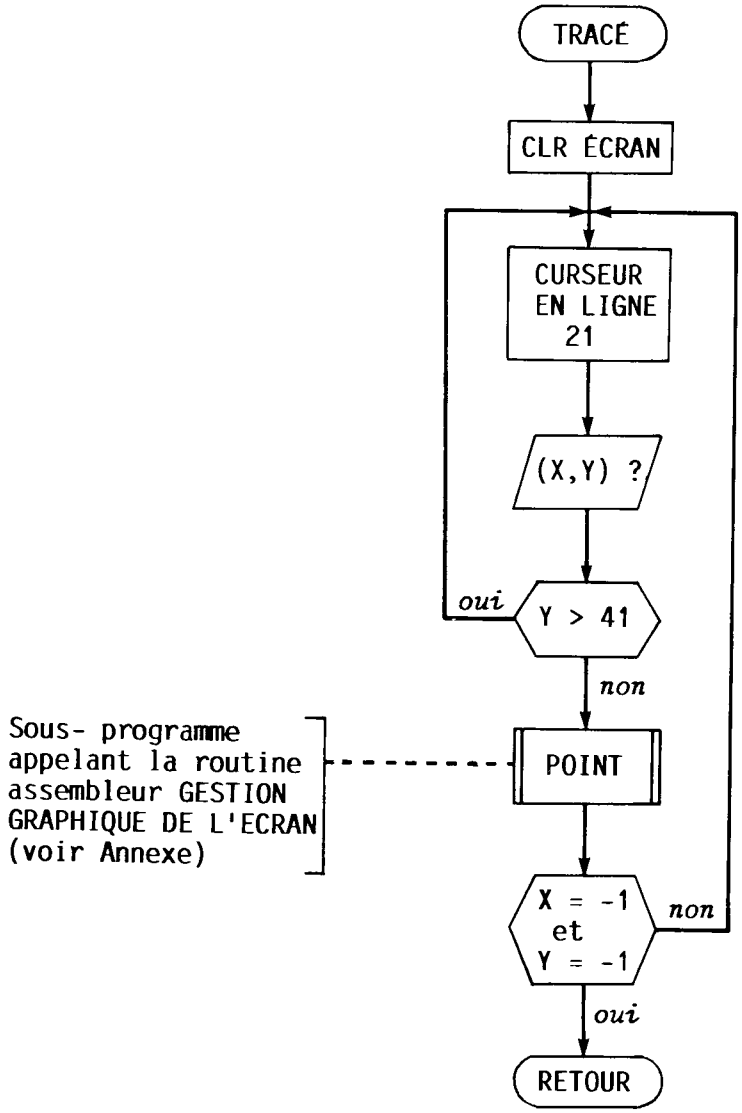
Coordonnées (X,Y) du point à allumer → POINT → *Affichage du point de coordonnées (X,Y)*

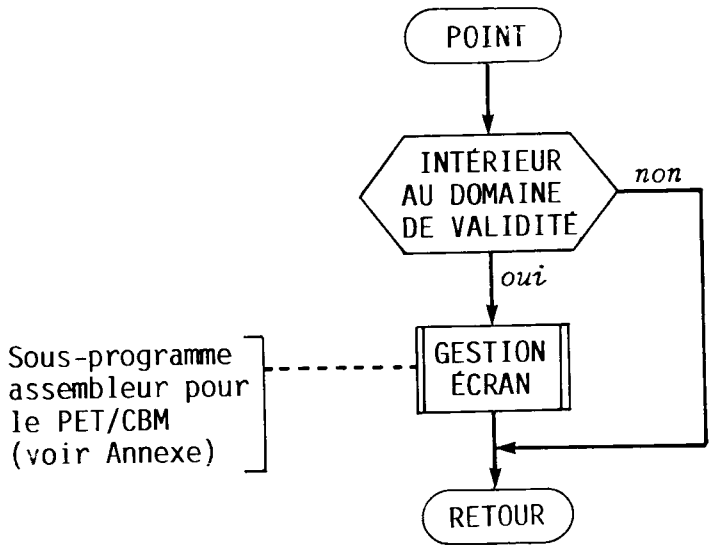
Structure



Organigrammes







Variables

POINT : Affichage d'un point sur l'écran

X : Abscisse du point à afficher
Y : Ordonnée du point à afficher
I : Indice de boucles

Particularités

- Le sous-programme ***POINT*** doit être modifié si l'on souhaite exécuter le programme sur des machines différentes.

- Dans le cas de l'APPLE, nous n'avons pas utilisé l'instruction POKE-16302,Ø, car nous voulions conserver le bénéfice des quatre lignes de texte à l'écran afin d'introduire les coordonnées des points à afficher.

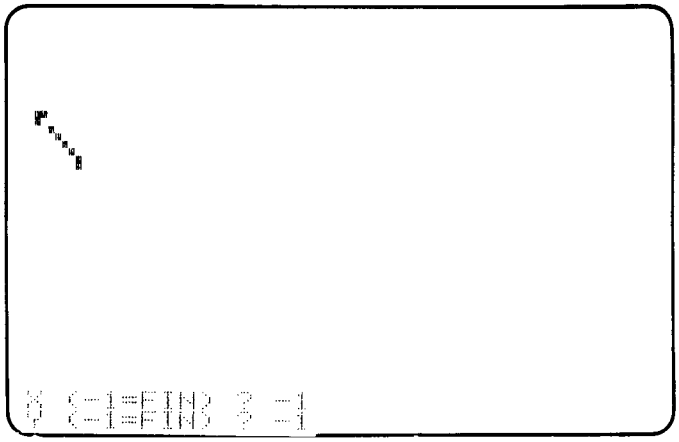
- Notons la présence de l'instruction HCOLOR=3 dans le programme destiné à l'APPLE. Elle permet un affichage en blanc sur fond noir en haute résolution.

LISTING APPLE

```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACE DE POINTS
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 4000: REM * TRACE *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
3000  REM ------------------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ---------------------------------------------------
3070  REM
3080  REM
4000  REM ------------------------------------------------------------
4002  REM TRACE
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE + TEXTE
4010  HGR : HCOLOR= 3
4015  REM EFFACEMENT ECRAN TEXTE
4020  HOME
4025  REM CURSEUR DANS LA FENETRE TEXTE
4030  VTAB 22: HTAB 1
4035  REM COORDONNEES DU POINT A TRACER
4040  INPUT "X (0-279,-1=FIN) ? ";X
4050  INPUT "Y (0-159,-1=FIN) ? ";Y
4060  GOSUB 3000: REM  * POINT *
4070  IF X <  >  - 1 OR Y <  >  - 1 THEN 4020
4080  HOME : TEXT
4090  RETURN : REM ---------------------------------------------------
4100  REM
```

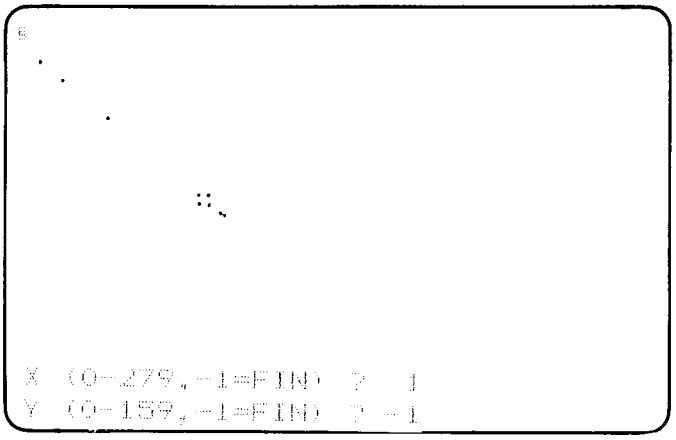
LISTING PET/CBM

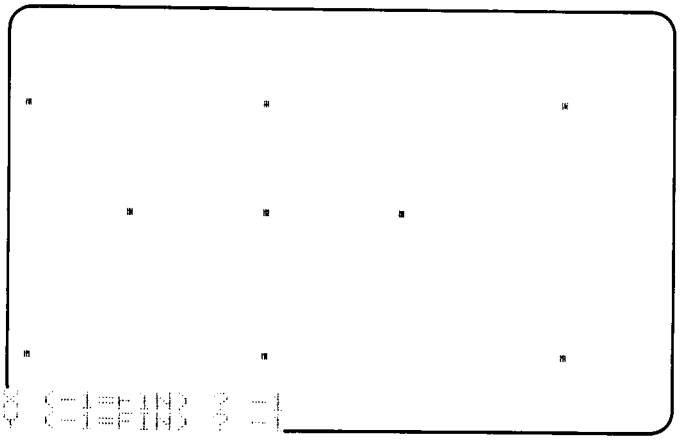
```
1000 REM------------------------------------------------------------
1002 REM*****              TRACE DE POINTS              *****
1004 REM------------------------------------------------------------
1010 :
1020 :GOSUB 4000 * TRACE *
1030 :
1040 END:------------------------------ PROG. PRINCIPAL ------
1050 :
1060 :
3000 REM------------------------------------------------------------
3002 REM*****                  POINT                    *****
3004 REM------------------------------------------------------------
3010 :REM TEST DE VALIDITE
3020 :       IF X<0 OR INT(X+.5)>79 THEN 3060
3030 :       IF Y<0 OR INT(Y+.5)>49 THEN 3060
3040 :REM AFFICHAGE DU POINT
3050 :       POKE975,INT(X+.5):POKE976,INT(Y+.5):SYS826
3060 RETURN:------------------------------------- POINT -----
3070 :
3080 :
4000 REM------------------------------------------------------------
4002 REM*****                  TRACE                    *****
4004 REM------------------------------------------------------------
4005 :REM EFFACEMENT ECRAN
4010 :       PRINT CHR$(147)
4015 :REM POSITION HOME
4020 :       PRINT CHR$(19)
4025 :REM ON PASSE 20 LIGNES
4030 :       FOR I=1 TO 20:PRINT:NEXT I
4035 :REM COORDONNEES DU POINT A TRACE
4040 :       INPUT"X (-1=FIN) ";X
4050 :       INPUT"Y (-1=FIN) ";Y
4055 :REM PROTECTION INPUT
4060 :       IF Y>41 THEN 4020
4070 :       GOSUB 3000 * POINT *
4080 :IF X<>-1 OR Y<>-1 THEN 4020
4090 RETURN:------------------------------------ TRACE ------
4100 :
```



PET/CBM exemple 1

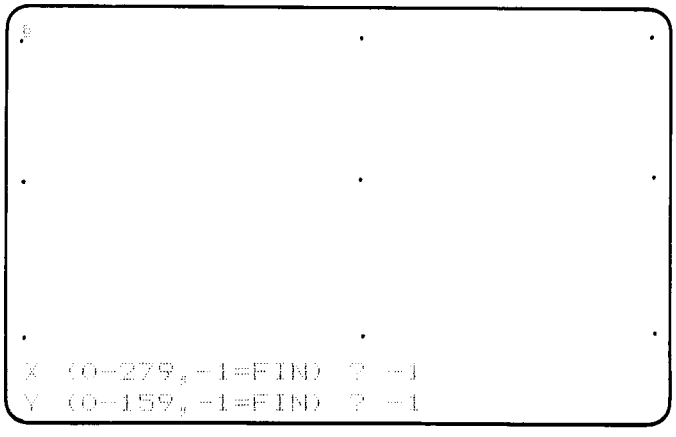
APPLE exemple 1





PET/CBM exemple 2

APPLE exemple 2



VECTEUR

A la fin de ce chapitre, nous serons en mesure de tracer des segments de droites sur l'écran que nous appellerons vecteurs. Ceux-ci, étant limités entre un point origine et un point arrivée, sont considérés comme un ensemble de points alignés et un vecteur sera constitué d'autant de points que la résolution de l'écran le permet et l'impose.

HYPOTHESE

Etant donnée notre conception d'un vecteur, tracer une telle représentation revient à allumer une suite finie de points. En conséquence, lorsque nous aurons défini cette suite, il ne nous restera plus qu'à allumer les points correspondants. En outre, c'est exactement la fonction réalisée par la routine ***POINT*** du chapitre précédent. Nous serons donc très facilement capables de réaliser une routine ***VECTEUR*** indépendante de la machine utilisée si nous mettons en oeuvre la routine ***POINT***. C'est ce que nous allons faire dans le cas général. Puis, nous allons particulariser ce sous-programme dans le cas de l'APPLE qui possède une instruction simple de tracé de vecteur.

MODELE

De nombreuses méthodes permettent de déterminer la suite de points à allumer pour tracer un vecteur, mais nous nous contenterons de celle qui consiste à paramétrer un segment de droite.

Cette méthode donne les valeurs successives des coordonnées **X** et **Y** des points à allumer. Après avoir fixé le point origine

(**V1,V2**) et le point arrivée (**V3,V4**) du vecteur, il suffit d'appliquer les formules suivantes en faisant varier la variable **T** entre 0 et 1 :

X = (V3-V1) * T + V1

Y = (V4-V2) * T + V2

pour obtenir tous les points du vecteur.

En fonction de la résolution de l'écran, il faut faire varier plus ou moins vite la variable **T** qui permet le calcul des coordonnées **X** et **Y**. Ainsi, pour les 4000 points de la résolution du PET/CBM, une variation avec un incrément de 0,1 est suffisante mais, avec une résolution, par exemple de 64000 points, il faudrait une valeur beaucoup plus faible.

Quel que soit l'incrément choisi, le principe du tracé d'un vecteur reste le même. La variable **T** commence par prendre la valeur Ø qui définit le premier point à allumer et dont les coordonnées sont en abscisse : **V1** et en ordonnée : **V2** . Ensuite, la variable T prend la valeur 0,1 qui permet de calculer le **ΔX** et le **ΔY** à ajouter respectivement à **V1** et **V2** pour déterminer les coordonnées du point suivant à allumer. Puis la variable **T** prend la valeur 0,2... Et ainsi de suite jusqu'à ce que **T** vaille 1 ; ce qui correspond à l'allumage du point de coordonnées **V3** et **V4**. A chaque calcul pour une nouvelle valeur de T l'appel de la routine POINT est effectué et tout naturellement, le vecteur est tracé.

Dans le cas de l'APPLE, il suffit d'utiliser l'instruction **HPLOT V1, V2 TO V3, V4** pour tracer un vecteur entre le point origine de coordonnées **V1,V2** et le point extrémité de coordonnées **V3, V4**. Pour des raisons de rapidité d'exécution, c'est l'instruction que nous utilisons dans le programme exemple, mais on aurait pu mettre en oeuvre la forme paramétrée vue ci-dessus qui donne d'ailleurs un tracé sensiblement plus fin sur l'écran.

EXEMPLE D'UTILISATION

Nous avons mis en oeuvre une routine de ***SAISIE DE COORDONNEES*** afin de pouvoir exploiter la routine ***VECTEUR***. Elle permet dans un premier temps d'introduire les coordonnées du premier point du vecteur, puis les coordonnées du second point et, dans un second temps, le tracé du vecteur ainsi défini, par appel de la routine ***VECTEUR***.

L'introduction des coordonnées des points définis ci-dessous permettent de se familiariser avec le tracé des vecteurs.

. **Premier vecteur :** Premier point X = 0 Y = 0
Second point X = 159 Y = 159

Le résultat de ce tracé se trouve en haut de la page de la page 29

. Second vecteur : Premier point X = 279 Y = 0
Second point X = 0 Y = 159

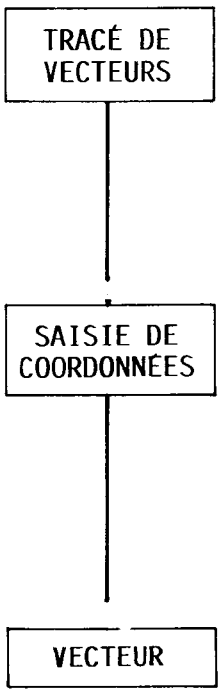
Le résultat de ce tracé se trouve en bas de la page de la page 29.

Pour sortir de l'exécution du programme, il suffit de taper un caractère 'F' après avoir tracé un vecteur.

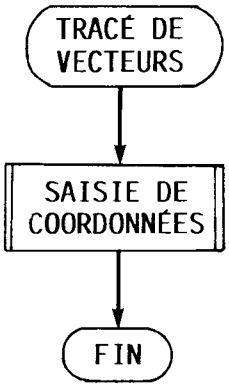
LE PROGRAMME

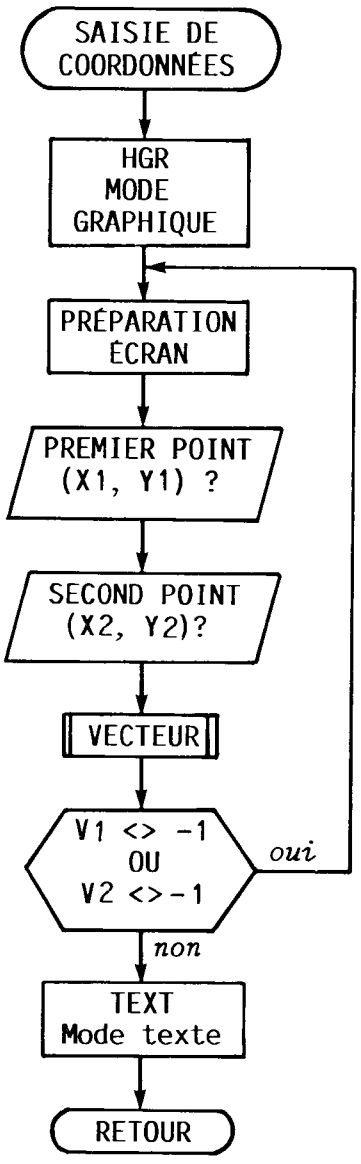
Traitement

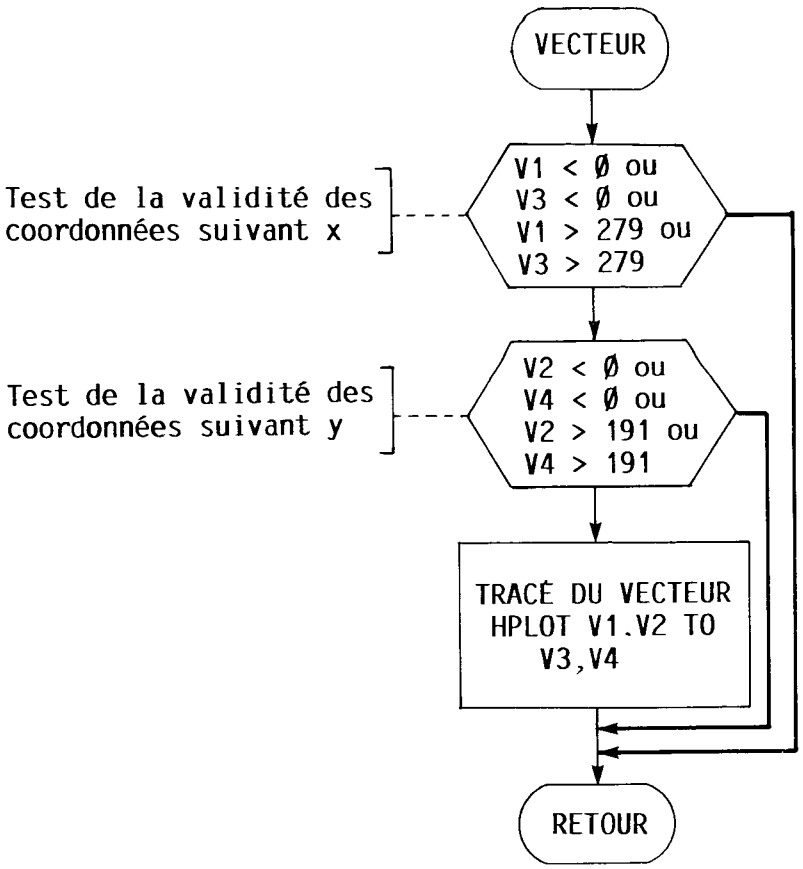
Structure



Organigrammes







Variables

VECTEUR : Tracé d'un vecteur.

V1, V2 : Coordonnées du premier point du vecteur à tracer.
V3, V4 : Coordonnées du second point du vecteur à tracer.

SAISIE DE COORDONNEES : Essais de tracés de vecteurs.

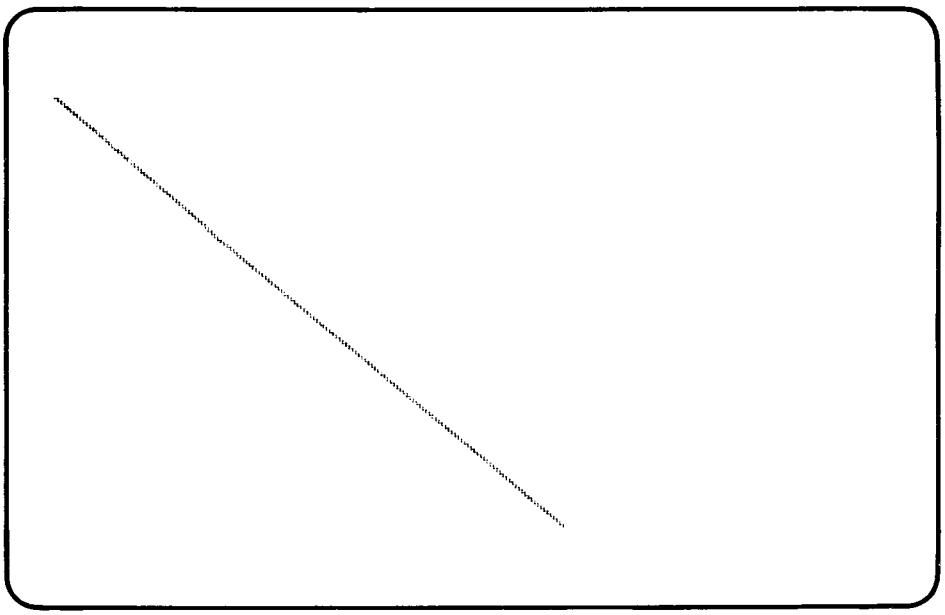
V1, V2 : Coordonnées du premier point du vecteur à tracer.
V2, V4 : Coordonnées du second point du vecteur à tracer.

Particularités

Le fait d'utiliser une méthode consistant à paramétrer les vecteurs à tracer est pénalisant pour les petits tracés. En effet, que nous tracions un vecteur qui traverse l'écran ou un vecteur qui n'occupe que la moitié de ce même écran, nous balayons toujours l'intervalle [0,1] à la même vitesse. Or, il faut deux fois plus de points pour traverser l'écran dans sa totalité plutôt que dans sa moitié. Cela revient à dire que quelle que soit la taille du vecteur, nous mettons toujours le même temps pour le tracer. Mais, le problème ne se pose pas sur APPLE étant donnée l'option que nous avons prise.

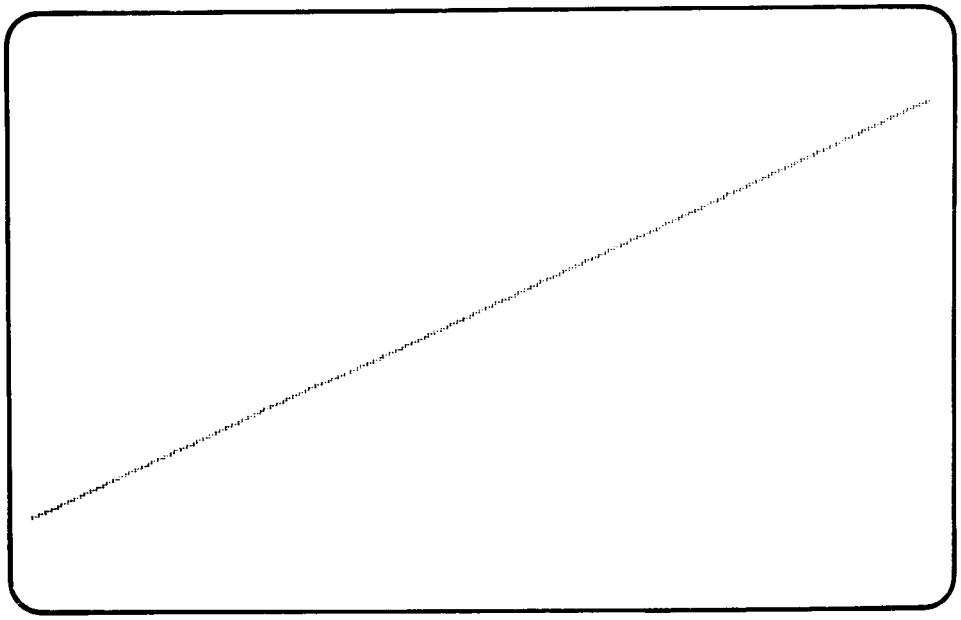
```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACE DE VECTEURS
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 5000: REM * SAISIE DE COORDONNEES *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM SAISIE DE COORDONNEES
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE + TEXTE
5010  HGR : HCOLOR= 3
5015  REM EFFACEMENT ECRAN
5020  HOME
5025  REM CURSEUR DANS LA FENETRE TEXTE
5030  VTAB 22: HTAB 1
5035  REM SAISIE ET TRACE
5040  INPUT "PREMIER POINT (X1,Y1) ? ";V1,V2
5050  INPUT "SECOND  POINT (X2,Y2) ? ";V3,V4
5060  GOSUB 4000: REM   * VECTEUR *
5070  IF V1 <  > - 1 OR V3 <  > - 1 THEN 5020
5080  HOME : TEXT
5090  RETURN : REM ---------------------------------------------------
5100  REM
```

Premier vecteur



```
PREMIER POINT (X1,Y1) ? 0,0
SECOND  POINT (X2,Y2) ? 159,159
```

Deuxième vecteur



```
PREMIER POINT (X1,Y1) ? 279,0
SECOND  POINT (X2,Y2) ? 0,159
```

POLYGONES

Après avoir considéré un vecteur comme un ensemble de points, nous allons tracer des polygones en les considérant comme un ensemble de vecteurs. Cependant, nous nous limiterons aux polygones à quatre angles dans ce chapitre.

HYPOTHESE

Notre but étant de relier quatre points quelconques de l'écran entre eux, il nous suffit de connaître l'ordre dans lequel il faut les relier, pour arriver à nos fins. En effet, une fois l'ordre défini, il suffira de les transmettre par couples à la routine ***VECTEUR*** pour qu'ils soient effectivement reliés ; sous-programme réalisé précédemment.

MODELE

Dans une première phase, nous allons mémoriser les quatre points des sommets du polygone que nous voulons tracer. Dans le programme, ceci est réalisé par les variables **P1** à **P8**. Chaque variable **Pi**, où i est impair, contient l'abscisse d'un point et, où i est pair, contient une ordonnée.

Lorsque les points sont définis, on aborde la seconde phase qui consiste à appeler quatre fois la routine ***VECTEUR*** en permutant, de façon circulaire, les couples de coordonnées des sommets, entre les variables V1 à V4, qui permettent de tracer des vecteurs.

En détaillant et en considérant connues les variables **P1** à **P8**, on trace d'abord le vecteur liant le premier point de coordonnées

(**P1, P2**) au second point de coordonnées (**P3, P4**). Ensuite, on trace le vecteur liant le second point de coordonnées (**P3,P4**) au troisième point de coordonnées (**P5, P6**)... Ainsi de suite, jusqu'à ce que l'on boucle le contour en traçant le vecteur qui mène de nouveau au premier point de coordonnées **P1, P2.**

C'est en employant cette méthode que le programme permet le tracé des polygones réguliers ou non, dont le nombre de côtés reste inférieur à quatre.

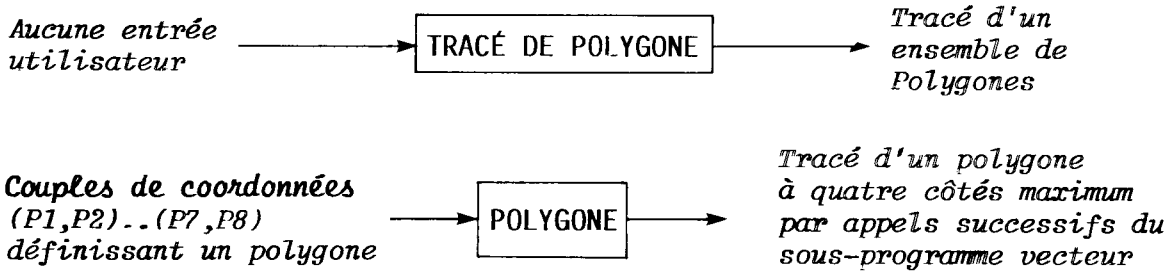
EXEMPLE D'UTILISATION

Plus qu'un exemple d'utilisation, le programme propose un ensemble de polygones choisis parmi les plus connus. Ainsi, nous trouvons sur l'écran, en partant du haut et de la gauche : un triangle, un carré, un rectangle, un parallélogramme, un trapèze.

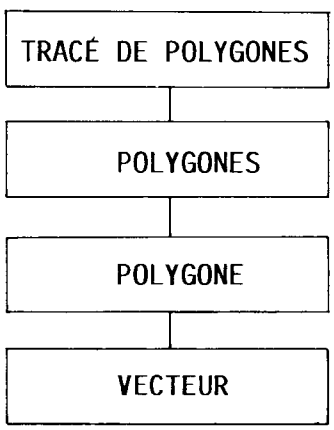
Bien entendu, si vous souhaitez vous familiariser avec le tracé de polygones, vous pouvez tenter de modifier les coordonnées des points et tracer des polygones, qu'ils soient réguliers ou non.

LE PROGRAMME

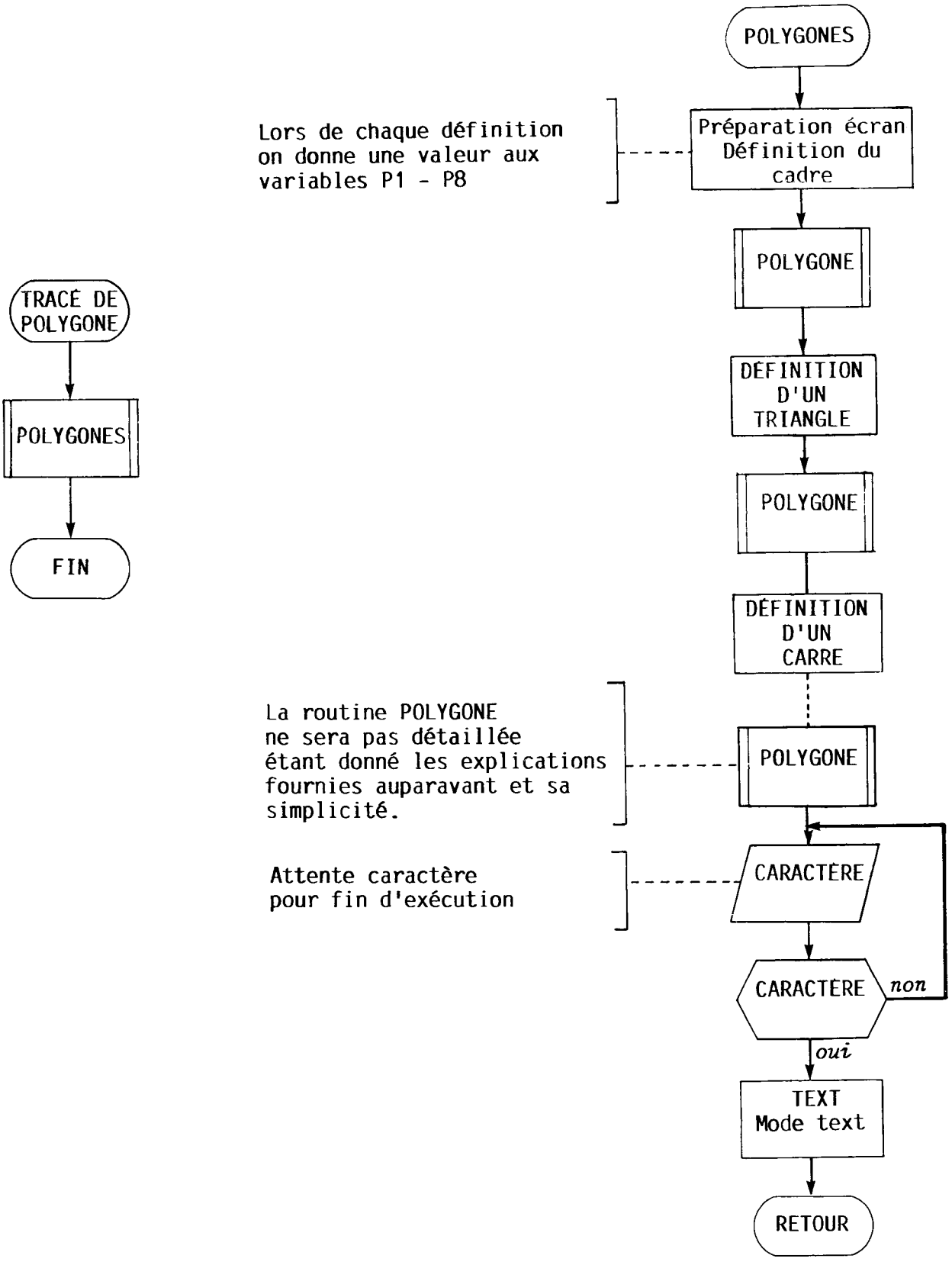
Traitement



Structure



Organigrammes



Variables

VECTEUR	:	Tracé d'un vecteur
V1,V2	:	Coordonnées du premier point du vecteur à tracer.
V3, V4	:	Coordonnées du second point du vecteur à tracer.
POLYGONE	:	Tracé d'un polygone
P1...P8	:	Coordonnées des points du polygone à tracer.
V1...V4	:	Coordonnées des points extrémités des vecteurs à tracer.
POLYGONES	:	Exemples de tracés de polygones.
P1...P8	:	Coordonnées des points du polygone à tracer.
R$	:	Saisie d'un caractère.

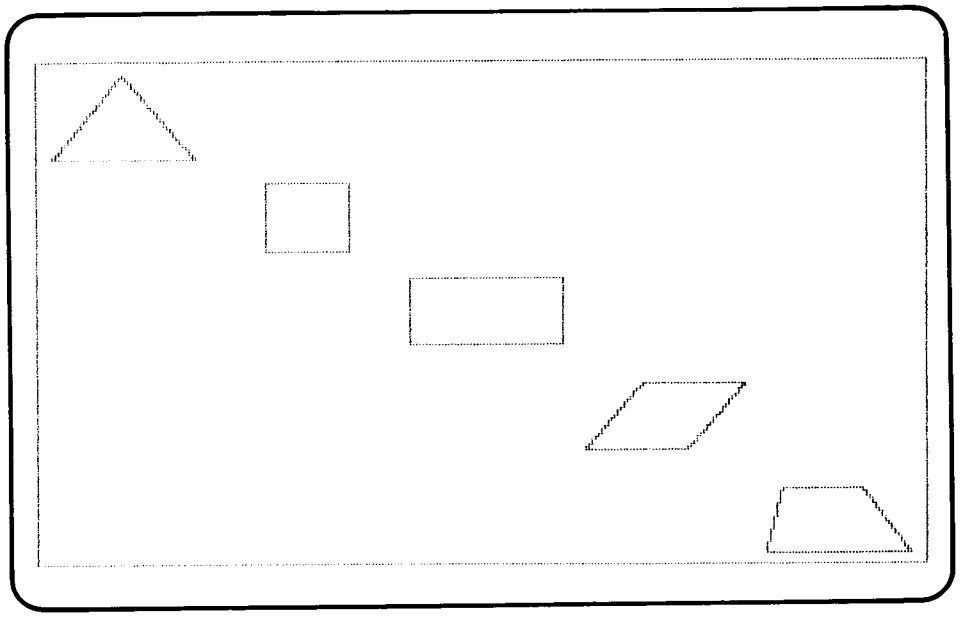
Particularités

- Dans sa version actuelle, le programme utilise les variables P1 à P8 pour mémoriser un polygone, or il faut noter que nous aurions pu utiliser un tableau de coordonnées pour obtenir le même effet. Dans certains cas d'utilisation, cette deuxième solution peut s'avérer plus satisfaisante.

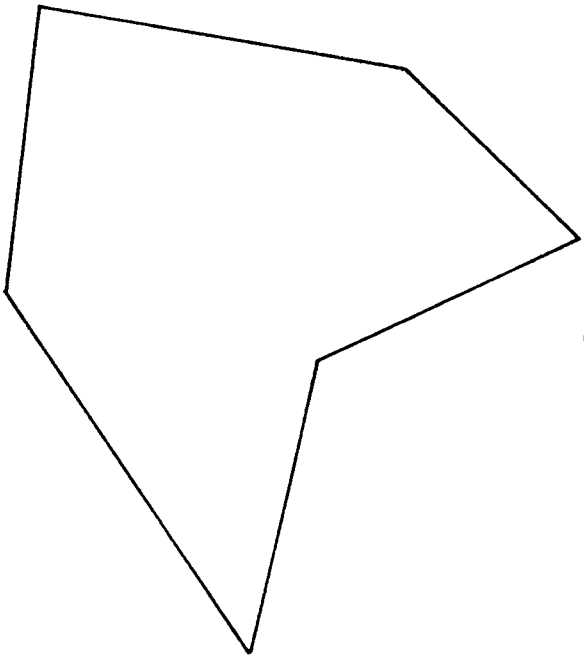
- Notons l'utilisation de l'instruction POKE-16302,Ø qui permet de passer en mode graphique plein écran.

```
1000  REM ------------------------------------------------------------
1002  REM TRACE DE POLYGONES
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 6000: REM * POLYGONES *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM POLYGONE
5004  REM ------------------------------------------------------------
5005  REM (P1,P2)...(P7,P8) : COUPLES DE COORDONNEES
5006  REM DES POINTS DU POLYGONE A TRACER
5010 V1 = P1:V2 = P2:V3 = P3:V4 = P4
5020  GOSUB 4000: REM *VECTEUR*
5030 V1 = P3:V2 = P4:V3 = P5:V4 = P6
5040  GOSUB 4000: REM * VECTEUR *
5050 V1 = P5:V2 = P6:V3 = P7:V4 = P8
5060  GOSUB 4000: REM * VECTEUR *
5070 V1 = P7:V2 = P8:V3 = P1:V4 = P2
5080  GOSUB 4000: REM * VECTEUR *
5090  RETURN : REM ---------------------------------------------------
5100  REM
5110  REM
```

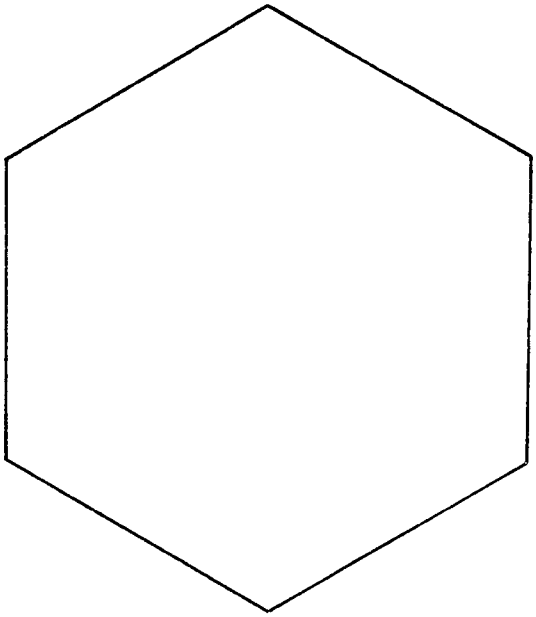
```
6000  REM ------------------------------------------------------------
6002  REM POLYGONES
6004  REM ------------------------------------------------------------
6005  REM PAGE 1 GRAPHIQUE
6010  HGR : POKE  - 16302,0: HCOLOR= 3
6020  REM EFFACEMENT ECRAN
6030  HOME
6035  REM CADRE
6040 P1 = 0:P2 = 0:P3 = 279:P4 = 0
6050 P5 = 279:P6 = 191:P7 = 0:P8 = 191
6060  GOSUB 5000: REM  * POLYGONE *
6065  REM TRIANGLE
6070 P1 = 5:P2 = 37:P3 = 50:P4 = 37
6080 P5 = 27:P6 = 5:P7 = 27:P8 = 5
6090  GOSUB 5000: REM  * POLYGONE *
6095  REM CARRE
6100 P1 = 72:P2 = 72:P3 = 98:P4 = 72
6110 P5 = 98:P6 = 46:P7 = 72:P8 = 46
6120  GOSUB 5000: REM  * POLYGONE *
6125  REM RECTANGLE
6130 P1 = 117:P2 = 82:P3 = 165:P4 = 82
6140 P5 = 165:P6 = 107:P7 = 117:P8 = 107
6150  GOSUB 5000: REM  * POLYGONE *
6155  REM PARALLELOGRAMME
6200 P1 = 190:P2 = 122:P3 = 222:P4 = 122
6210 P5 = 204:P6 = 147:P7 = 172:P8 = 147
6220  GOSUB 5000: REM  * POLYGONE *
6225  REM TRAPEZE
6230 P1 = 234:P2 = 162:P3 = 259:P4 = 162
6240 P5 = 274:P6 = 187:P7 = 229:P8 = 187
6250  GOSUB 5000: REM  * POLYGONE *
6260  GET R$: IF R$ = "" THEN 6260
6270  TEXT
6280  RETURN : REM ------------------------------------------------------
6290  REM
```



POLYGONE RÉGULIER



et



sont deux polygones mais, dans ce chapitre nous ne nous intéresserons qu'au second car, il est régulier, ses côtés sont de même longueur, et il est inscrit dans un cercle.

Nous allons réaliser un sous-programme permettant de tracer des polygones du même type que celui qui nous intéresse ceci, quels que soient la longueur et le nombre des côtés.

HYPOTHESE

Les polygones, nous l'avons vu dans le chapitre précédent, sont des contours constitués de suites de segments de droite ; nous allons donc, une fois encore, utiliser la routine de base permettant de tracer des vecteurs. Ces derniers organisés astucieusement par un sous-programme que nous appellerons ***POLYGONE REGULIER,***

nous permettront alors d'atteindre notre objectif sans plus de difficultés.

MODELE

Les polygones que nous allons tracer sont inscrits dans un cercle dont nous allons, en premier lieu, localiser le centre. Cette opération est effectuée en fixant des coordonnées écran dans les variables **P1** et **P2**. Ensuite, parce qu'un cercle ne serait pas un cercle si l'on ne définissait pas son rayon, nous fixons deux nouvelles coordonnées dans les variables **P3** et **P4** de la routine ***POLYGONE REGULIER***. Ces deux points définissent ce que nous appellerons le segment directeur dont l'extrémité nous servira de point de départ au tracé du polygone dont il représente l'un des sommets.

Les côtés du polygone régulier sont des vecteurs, or, rappelons nous, pour tracer un vecteur, il faut deux points, soit quatre coordonnées. Si, au début, les coordonnées du premier point sont connues puisqu'elles correspondent à celles du sommet de départ fixé par l'utilisateur, il faut cependant calculer celles du second. Ce calcul est pris en charge par le sous-programme.

C'est en utilisant l'équation paramétrique d'un cercle de rayon égal à la longueur du segment directeur (**C1** en projection sur l'axe des abscisses égal à la différence entre **P3** et **P1**, **C2** en projection sur l'axe des ordonnées égal à la différence entre les variables **P4** et **P2** du sous-programme ***POLYGONE REGULIER***), **de centre (P1;P2) et, en effectuant un calcul tous les 2π/NC**, où **NC** est la dernière caractéristique fournie par l'utilisateur et représente le nombre de côtés du polygone, **que les différents points extrémités des vecteurs sont déterminés.** A chaque calcul, le dernier point déterminé sert de point origine pour le vecteur à tracer et, en balayant l'intervalle **2*π**, c'est-à- dire en parcourant tout le cercle, nous sommes en mesure de réaliser l'ensemble du polygone.

EXEMPLE D'UTILISATION

L'exemple proposé trace deux polygones comme vous pouvez le voir page 40. Le polygone intérieur est décrit dans le programme comme suit :

P1 = 140 : P2 = 96 : P3 = 190 : P4 = 96 : NC = 6

cela signifie que le cercle dans lequel il est inscrit a pour centre le point de coordonnées **X = 140, Y = 96** et un rayon qui va du centre au point de coordonnées **X = 190, Y = 96**. Ce dernier est aussi, comme vous pouvez le vérifier lors de l'exécution, le premier des **NC = 6** sommets.

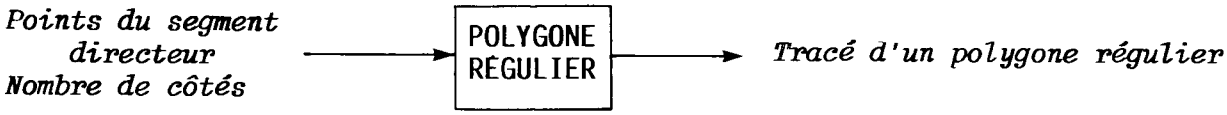
Le polygone extérieur, quant à lui, est inscrit dans un cercle de même centre, il a un segment directeur plus grand que le poly-

gone intérieur, et pour cause, il possède 8 côtés.

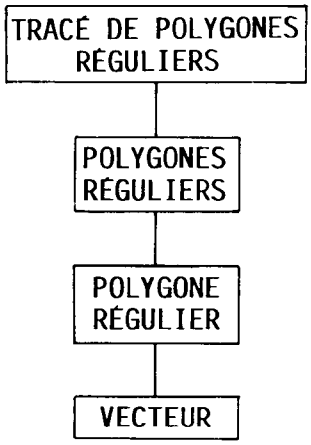
Nous aurions pu, comme vous allez certainement le faire, multiplier les exemples de polygones réguliers...

LE PROGRAMME

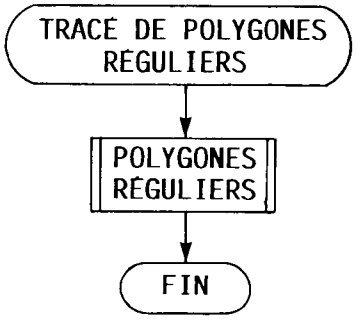
Traitement

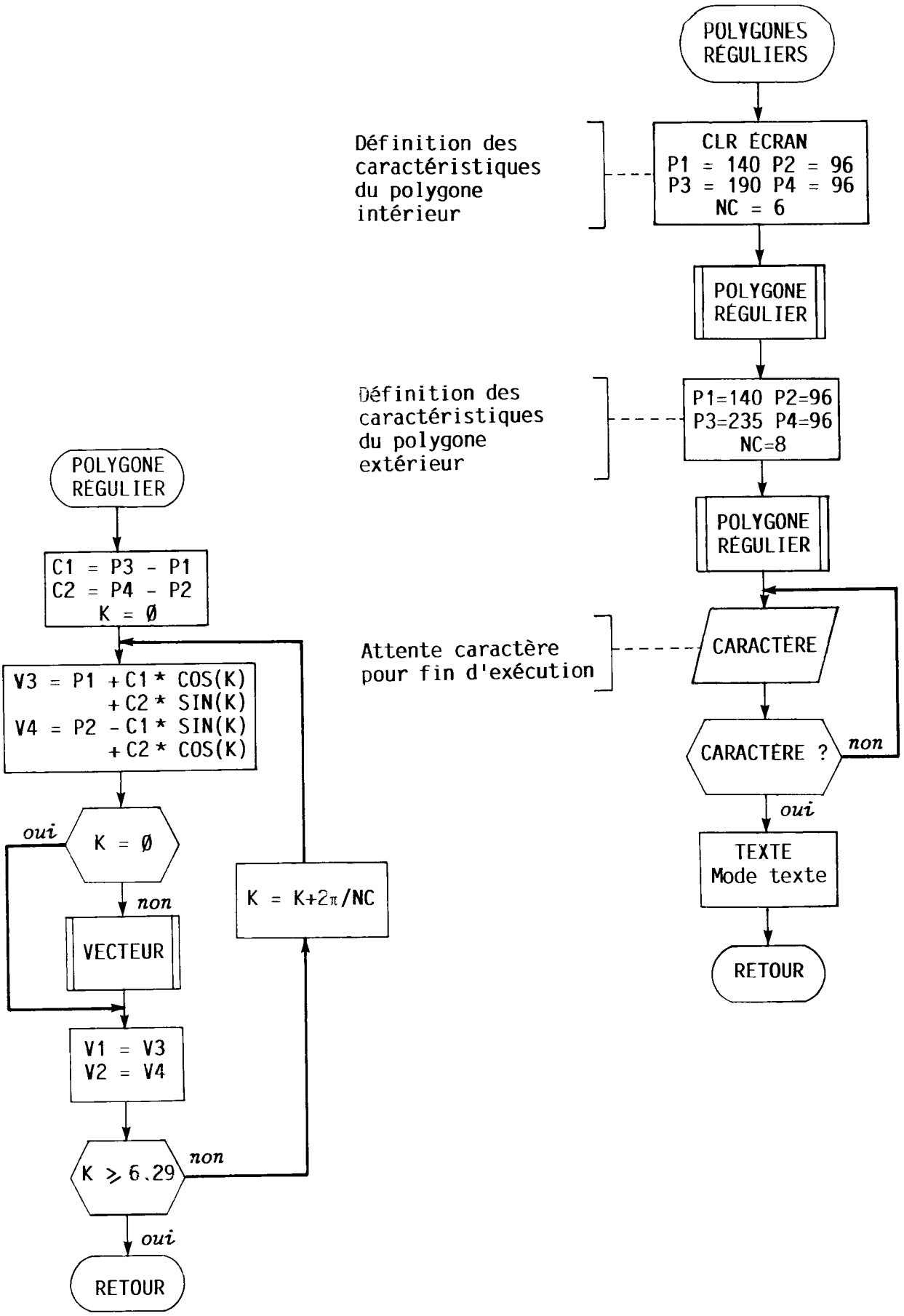


Structure



Organigrammes




POLYGONES RÉGULIERS
Définition des caractéristiques du polygone intérieur
CLR ÉCRAN
P1 = 140 P2 = 96
P3 = 190 P4 = 96
NC = 6
POLYGONE RÉGULIER
Définition des caractéristiques du polygone extérieur
P1=140 P2=96
P3=235 P4=96
NC=8
POLYGONE RÉGULIER
Attente caractère pour fin d'exécution
CARACTÈRE
CARACTÈRE ?
non
oui
TEXTE
Mode texte
RETOUR
POLYGONE RÉGULIER
C1 = P3 - P1
C2 = P4 - P2
K = Ø
V3 = P1 + C1 * COS(K)
+ C2 * SIN(K)
V4 = P2 - C1 * SIN(K)
+ C2 * COS(K)
oui
K = Ø
non
VECTEUR
K = K+2π/NC
V1 = V3
V2 = V4
K ≥ 6.29
non
oui
RETOUR

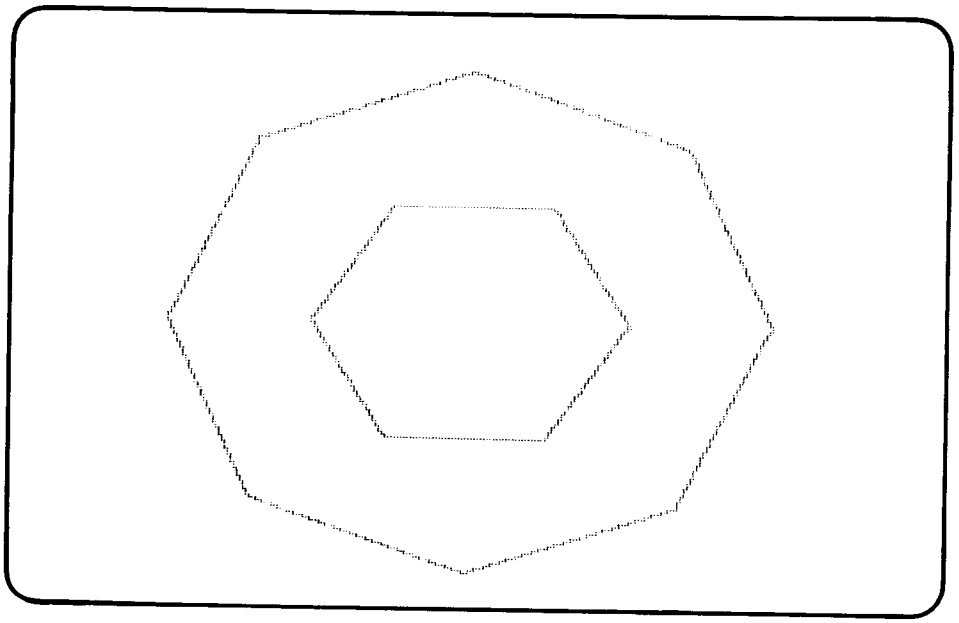
Variables

VECTEUR : Tracé d'un vecteur.
V1,V2 : Coordonnées du premier point du vecteur à tracer.
V3,V4 : Coordonnées du second point du vecteur à tracer.

POLYGONE REGULIER : Tracé d'un polygone régulier.
P1,P2 : Coordonnées du centre du cercle circonscrit au polygone régulier.
P3,P4 : Coordonnées du premier sommet du polygone régulier.
NC : Nombre de côtés du polygone régulier.
C1 : Projection du segment directeur sur l'axe des abscisses.
C2 : Projection du segment directeur sur l'axe des ordonnées.
K : Incrément de calcul des coordonnées des sommets du polygone régulier.
V1,V2,V3, V4 : Coordonnées des vecteurs à tracer.

POLYGONES REGULIERS : Exemples des vecteurs à tracer.
P1...P4 : Coordonnées des segments directeurs.
NC : Nombre de côtés.
R$: Caractère pour fin d'exécution.

Exemple de tracé de polygone régulier



```
§LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACE DE POLYGONES REGULIERS
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 6000: REM * POLYGONES REGULIERS *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM POLYGONE REGULIER
5004  REM ------------------------------------------------------------
5005  REM (P1,P2),(P3,P4) : POINTS DU SEGMENT DIRECTEUR
5006  REM NC : NOMBRE DE COTES
5010 C1 = P3 - P1:C2 = P4 - P2
5020  FOR K = 0 TO 6.29 STEP 6.28 / NC
5030 V3 = P1 + C1 *  COS (K) + C2 *  SIN (K)
5040 V4 = P2 - C1 *  SIN (K) + C2 *  COS (K)
5050  IF K <  > 0 THEN  GOSUB 4000: REM * VECTEUR *
5060 V1 = V3:V2 = V4
5070  NEXT K
5080  RETURN : REM ---------------------------------------------------
5090  REM
5100  REM
6000  REM ------------------------------------------------------------
6002  REM POLYGONES REGULIERS
6004  REM ------------------------------------------------------------
6005  REM PAGE 1 GRAPHIQUE
6010  HGR : POKE  - 16302,0: HCOLOR= 3
6015  REM POLYGONE INTERIEUR
6020 P1 = 140:P2 = 96:P3 = 190:P4 = 96:NC = 6
6030  GOSUB 5000: REM  * POLYGONE REGULIER *
6035  REM POLYGONE EXTERIEUR
6040 P1 = 140:P2 = 96:P3 = 235:P4 = 96:NC = 8
6050  GOSUB 5000: REM  * POLYGONE REGULIER *
6060  GET R$: IF R$ = "" THEN 6060
6070  HOME : TEXT
6080  RETURN : REM ---------------------------------------------------
6090  REM
```

ELLIPSE

Jusqu'à présent nous avons vu des sous-programmes permettant de tracer des contours polygonaux et chaque fois nous avons été amenés à utiliser le sous-programme ***VECTEUR.*** Dans ce chapitre nous allons aborder un autre type de représentation qui va nous permettre, avec une méthode identique à celle utilisée précédemment, de tracer des cercles et des ellipses.

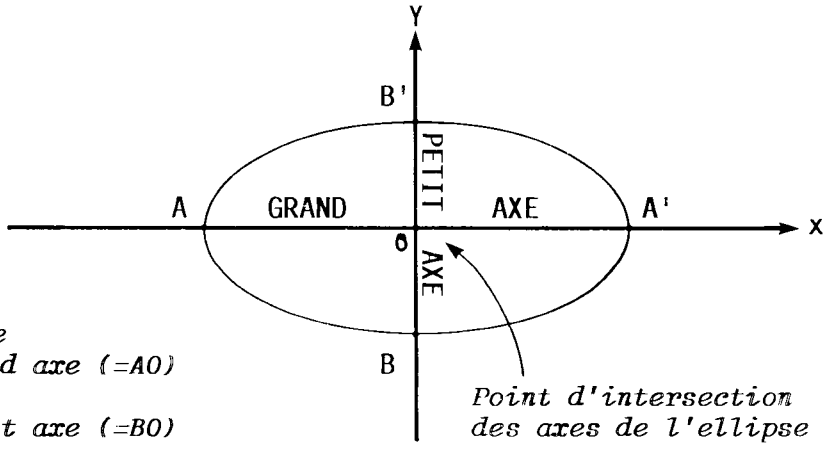
HYPOTHESE

Si nous n'utilisons plus la routine permettant de tracer des vecteurs, représenter une figure nécessite toujours l'affichage des points et en conséquence, nous utiliserons le sous-programme ***POINT.*** Celui-ci n'est pas utilisé pour tracer des vecteurs sur APPLE mais, rappelons que sur certains PSI il est présent dans tous les programmes de tracés.

MODELE

Afin de tracer une ellipse ou un cercle, nous utilisons l'équation paramétrique d'une telle représentation. Il est alors très simple de tracer une ellipse car il suffit de définir trois paramètres et d'appliquer des équations.

Définissons les noms et les points importants d'une ellipse sur le dessin suivant :



AA'=Longueur du grand axe
OA'=Longueur du demi grand axe (=AO)
BB'=Longueur du petit axe
OB'=Longueur du demi petit axe (=BO)

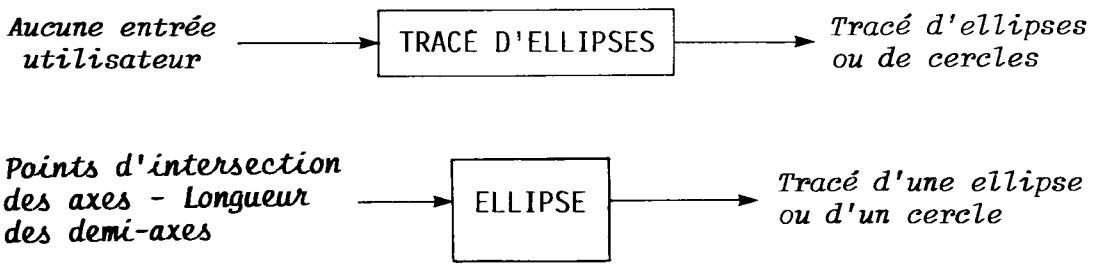
Pour utiliser la routine ***ELLIPSE*** qui comme son nom l'indique permet de tracer des ellipses, il suffit de fixer les coordonnées écran du point d'intersection des axes dans les variables **E1** et **E2** ainsi que les demi-longueurs des axes de l'ellipse à tracer dans les variables **E3** et **E4.** Comme nous l'avons déjà noté, la méthode employée pour tracer des ellipses permet aussi de tracer des cercles. Pour cela, il suffit d'effectuer la même valeur, de ce qui ne s'appelle plus axe mais rayon, aux variables **E3** et **E4.**

EXEMPLE D'UTILISATION

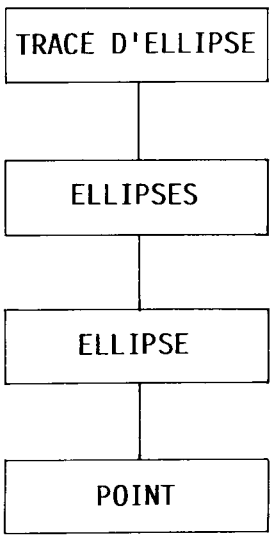
Le programme propose le tracé d'une ellipse et d'un cercle que vous pouvez voir page 47.

Nous constatons, que même si l'on voit clairement que les contours tracés ne sont pas des polygones, nous pourrions douter que la routine ***VECTEUR*** soit inutilisée étant donnée la faible résolution de l'écran. Pourtant la méthode employée et exposée ci-dessus, prouve que la qualité d'un segment est obtenue uniquement en fonction de la précision du tracé.

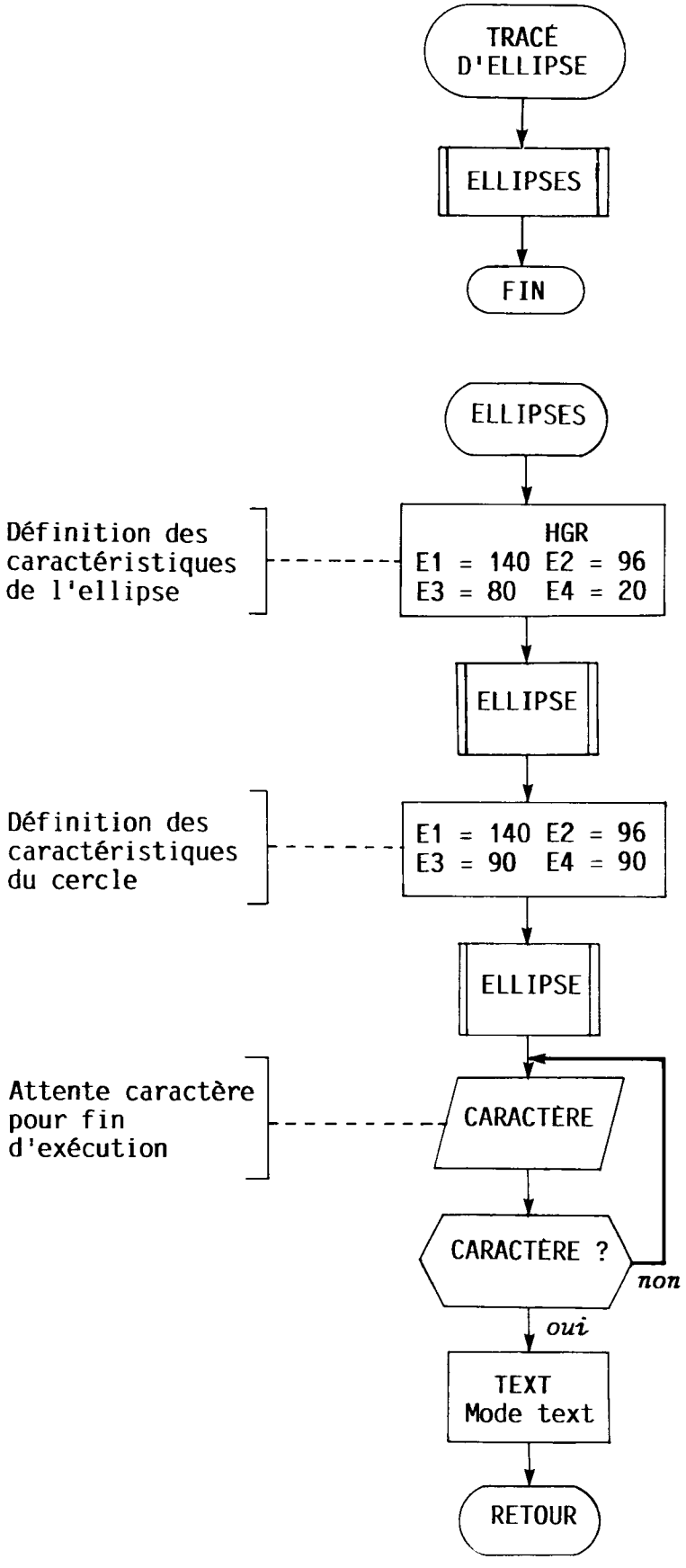
Traitement

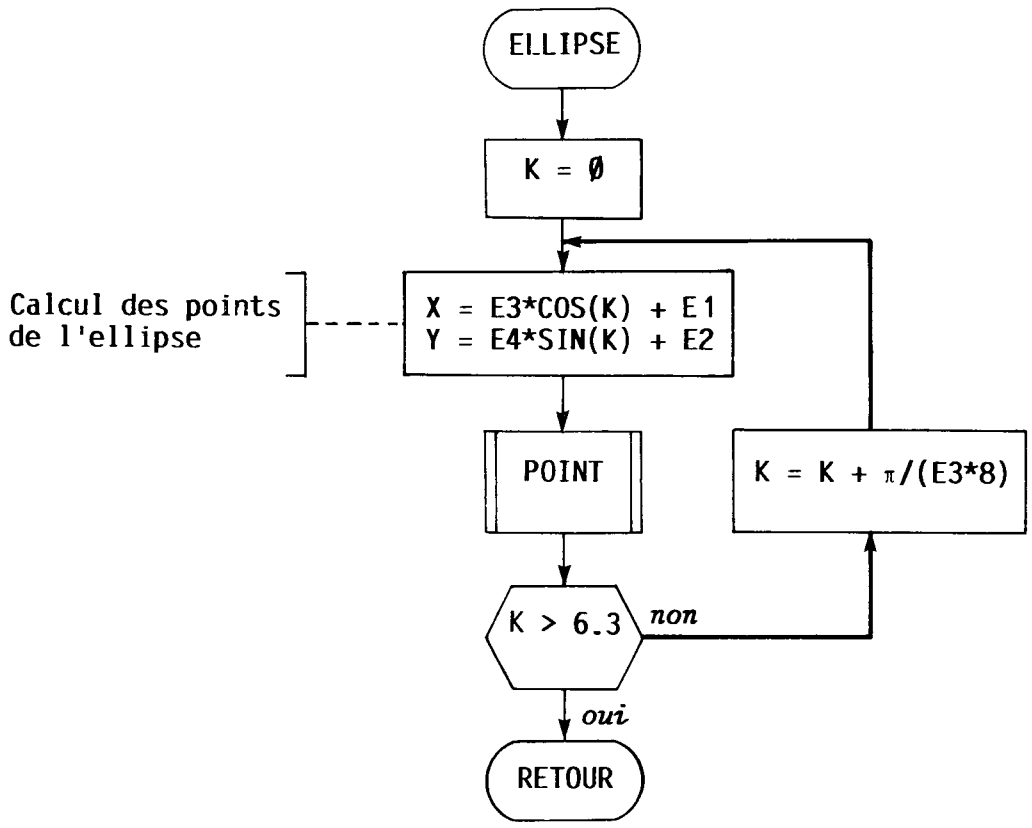


Structure



Organigrammes

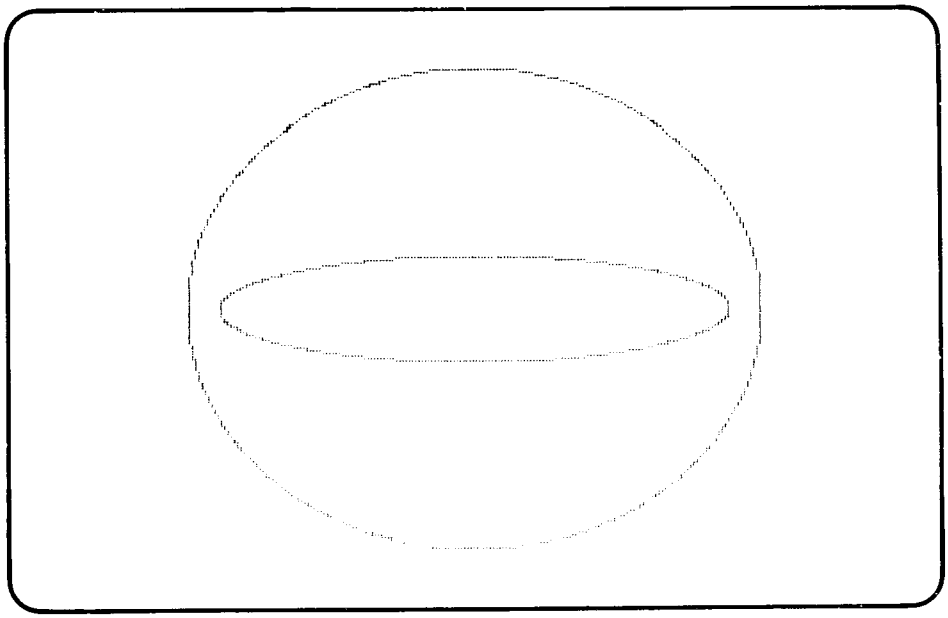




Variables

POINT	:	Affichage d'un point.
X	:	Abscisse du point à afficher.
Y	:	Ordonnée du point à afficher.
ELLIPSE	:	Tracé d'une ellipse.
E1,E2	:	Coordonnées du point d'intersection des axes de l'ellipse.
E3,E4	:	Longueurs des demi-axes de l'ellipse.
X,Y	:	Coordonnées des points de l'ellipse à afficher.
K	:	Incrément de calcul des points de l'ellipse.
ELLIPSES	:	Tracé d'un ensemble d'ellipses.
E1,E2,E3,E4	:	Caractéristiques des ellipses à tracer.
R$	:	Caractère pour fin d'exécution.

```
]LIST1000-

1000  REM ----------------------------------------------------------
1002  REM TRACE D'ELLIPSES
1004  REM ----------------------------------------------------------
1010  REM
1020  GOSUB 5000: REM * ELLIPSES *
1030  REM
1040  END : REM ----------------------------------------------------
1050  REM
1060  REM
3000  REM ----------------------------------------------------------
3002  REM POINT
3004  REM ----------------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM -------------------------------------------------
3070  REM
3080  REM
4000  REM ----------------------------------------------------------
4002  REM ELLIPSE
4004  REM ----------------------------------------------------------
4005  REM (E1,E2) : INTERSECTION DES AXES
4006  REM  E3,E4  : LONGUEUR DES DEMI AXES
4010  FOR K = 0.01 TO 6.3 STEP 6.28 / (E3 * 8)
4020 X = E3 *  COS (K) + E1
4030 Y = E4 *  SIN (K) + E2
4040  GOSUB 3000: REM * POINT *
4050  NEXT K
4060  RETURN : REM -------------------------------------------------
4070  REM
4080  REM
5000  REM ----------------------------------------------------------
5002  REM ELLIPSES
5004  REM ----------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM ELLIPSE INTERIEURE
5020 E1 = 140:E2 = 96:E3 = 80:E4 = 20
5030  GOSUB 4000: REM * ELLIPSE *
5035  REM ELLIPSE EXTERIEURE
5040 E1 = 140:E2 = 96:E3 = 90:E4 = 90
5050  GOSUB 4000: REM * ELLIPSE *
5060  GET R$: IF R$ = "" THEN 5060
5070  HOME : TEXT
5080  RETURN : REM -------------------------------------------------
5090  REM
```



Variations sur les tracés graphiques de base

POINTS ALÉATOIRES

Afficher des points aléatoirement sur l'écran donne parfois de très agréables effets graphiques et c'est la première représentation que nous verrons à partir des tracés graphiques de base.

MODELE

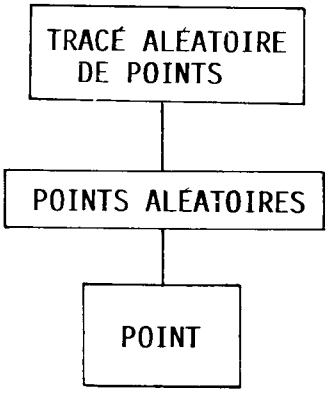
Les résultats d'exécution de la page 52 sont très simples à obtenir. Disposant d'une fonction **RND** générant des nombres aléatoires compris entre 0 et 1, il suffit de prendre le résultat de la multiplication du nombre aléatoire généré par la longueur maximale utile sur l'axe des abscisses d'une part et, le résultat de la multiplication d'un autre nombre aléatoire par la longueur maximale utile sur l'axe des ordonnées d'autre part. Ces résultats sont alors utilisés comme coordonnées des points à allumer sur l'écran.

En répétant ces calculs 1000 fois, devinez combien de points sont affichés. Si vous ne trouvez pas, il ne vous reste plus qu'à les compter sur les deux exemples d'exécution de la page 52.

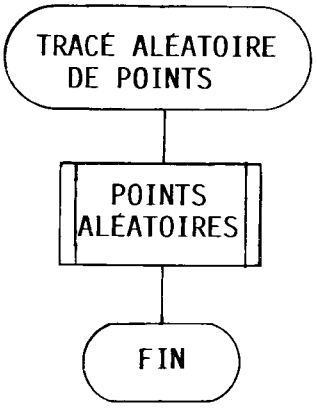
PROGRAMME

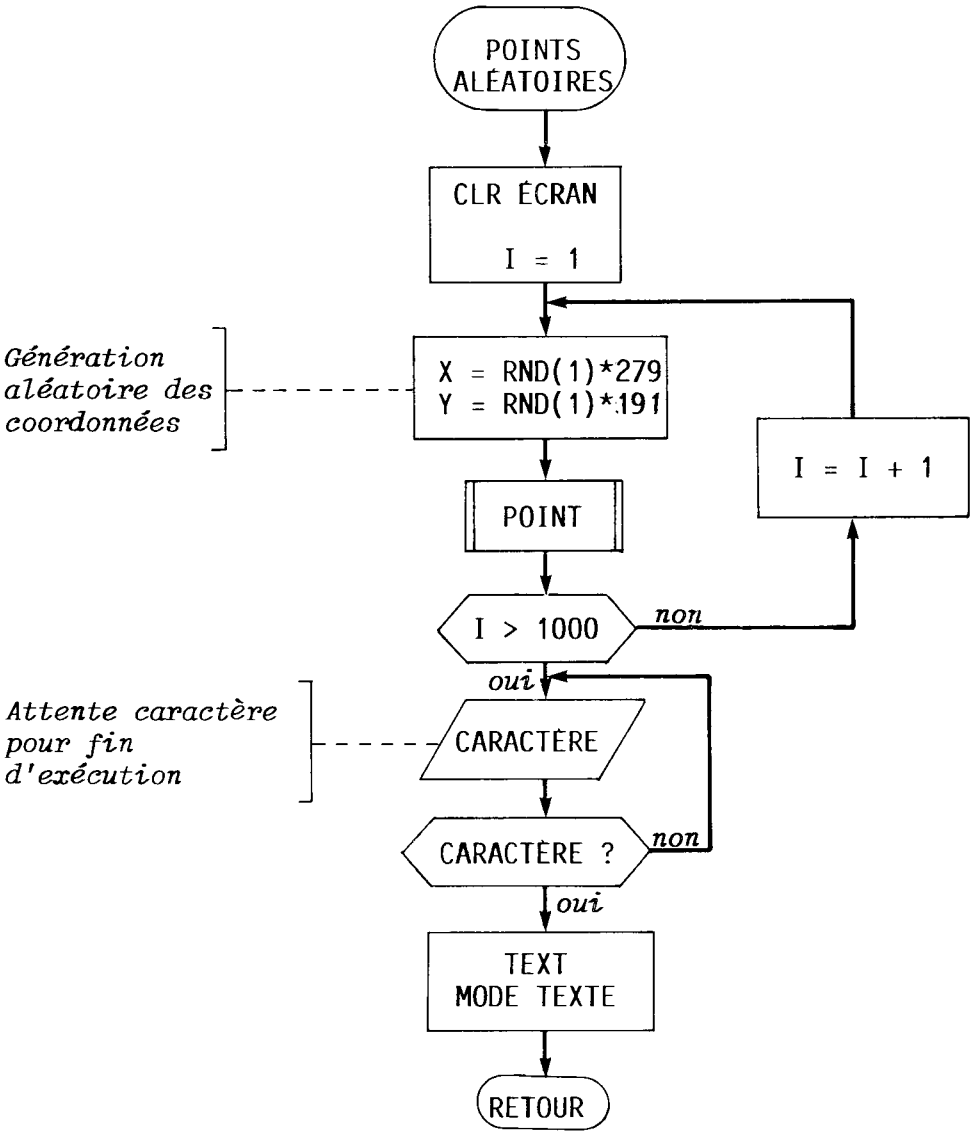
Traitement

Structure



Organigrammes




POINTS ALÉATOIRES
CLR ÉCRAN
I = 1
Génération aléatoire des coordonnées
X = RND(1)*279
Y = RND(1)*191
POINT
I = I + 1
I > 1000
non
oui
Attente caractère pour fin d'exécution
CARACTÈRE
CARACTÈRE ?
non
oui
TEXT
MODE TEXTE
RETOUR

Variables

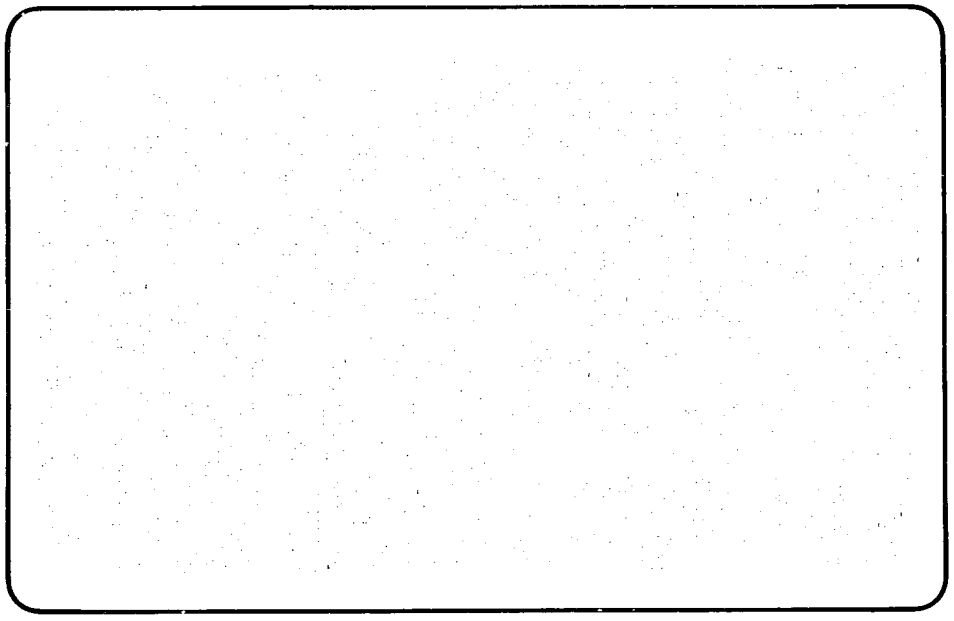
POINT : Affichage d'un point.
X : Abscisse du point à afficher.
Y : Ordonnée du point à afficher.

POINTS ALEATOIRES :

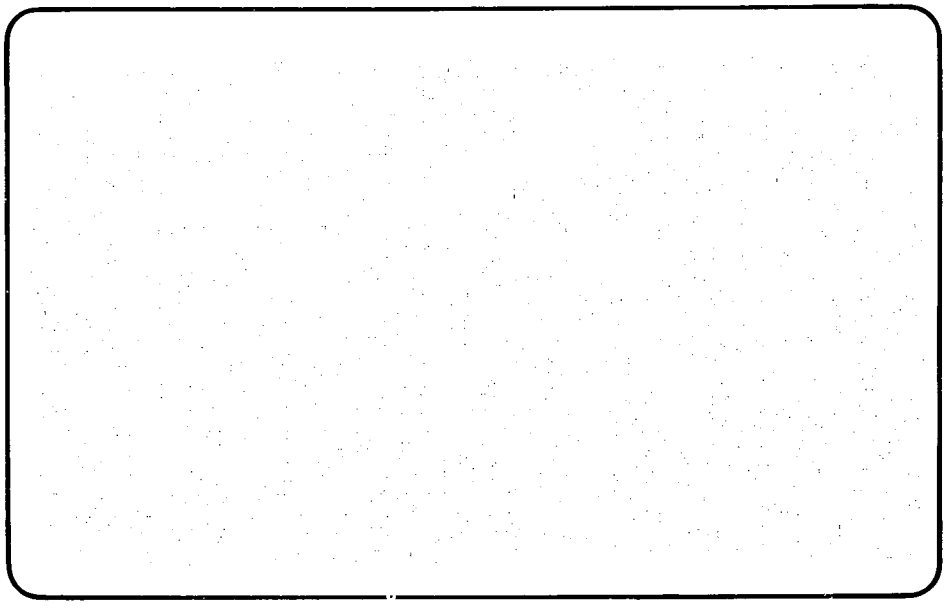
I : Compteur du nombre de points à afficher.
X,Y : Coordonnées du point à afficher.
R$: Caractère pour fin d'exécution.

```
$LIST1000-

1000  REM ------------------------------------------------
1002  REM TRACE ALEATOIRE DE POINTS
1004  REM ------------------------------------------------
1010  REM
1020  GOSUB 4000: REM * POINTS ALEATOIRES *
1030  REM
1040  END : REM ------------------------------------------
1050  REM
1060  REM
3000  REM ------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ----------------------------------------
3070  REM
3080  REM
4000  REM ------------------------------------------------
4002  REM POINTS ALEATOIRES
4004  REM ------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM TRACE DE POINTS ALEATOIRES
4020  FOR I = 1 TO 1000
4030 X =  RND (1) * 279
4040 Y =  RND (1) * 191
4050  GOSUB 3000: REM * POINT *
4060  NEXT I
4070  GET R$: IF R$ = "" THEN 4070
4080  HOME : TEXT
4090  RETURN : REM ----------------------------------------
4100  REM
```

Exemple de tracé aléatoire 1



Exemple de tracé aléatoire 2



VECTEURS ALÉATOIRES

Ce qui est possible pour l'affichage des points l'est aussi pour les vecteurs et dans ce chapitre, nous allons organiser des points pour former des segments de droite que nous allons disposer aléatoirement sur l'écran.

MODELE

Pour tracer un vecteur, il faut deux points, soit quatre coordonnées. Ces dernières sont calculées à partir de la génération de nombres aléatoires, ce qui permet de positionner un ensemble de vecteurs dont l'organisation dépend uniquement du hasard.

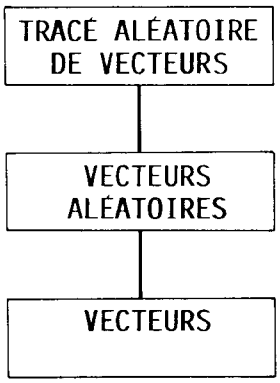
Cette méthode, très simple à programmer comme le montre le programme de ce chapitre, donne des effets d'une esthétique souvent remarquable.

D'autres variations sont possibles à partir d'un générateur de nombres aléatoires appliqué au tracé de vecteurs. Ainsi, on peut, comme la routine ***VECTEURS ALEATOIRES 2*** le propose, fixer une ou plusieurs coordonnées et faire varier les autres. Les effets obtenus sont tout aussi étonnants. D'ailleurs, pour vous en convaincre, il vous suffit de modifier la routine ***VECTEURS ALEATOIRES 1***, comme il a été fait dans la routine ***VECTEURS ALEATOIRES 2***, selon votre propre inspiration.

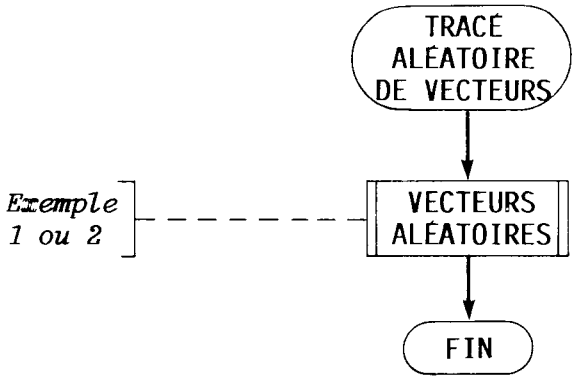
LE PROGRAMME

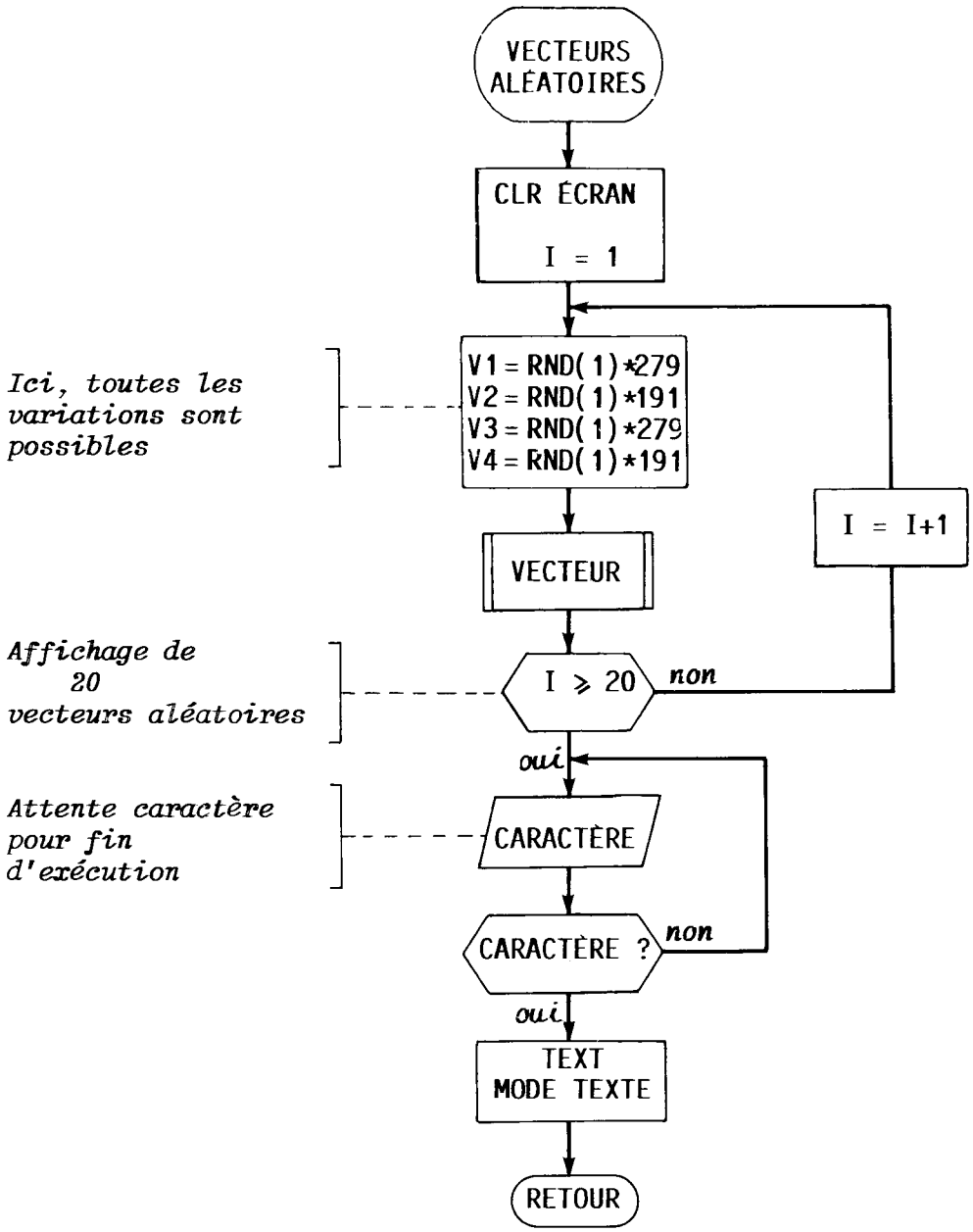
Traitement

Structure



Organigrammes




VECTEURS ALÉATOIRES
CLR ÉCRAN
I = 1
V1 = RND(1)*279
V2 = RND(1)*191
V3 = RND(1)*279
V4 = RND(1)*191
Ici, toutes les variations sont possibles
VECTEUR
I = I+1
Affichage de 20 vecteurs aléatoires
I ⩾ 20
non
oui
Attente caractère pour fin d'exécution
CARACTÈRE
CARACTÈRE ?
non
oui
TEXT
MODE TEXTE
RETOUR

Variables

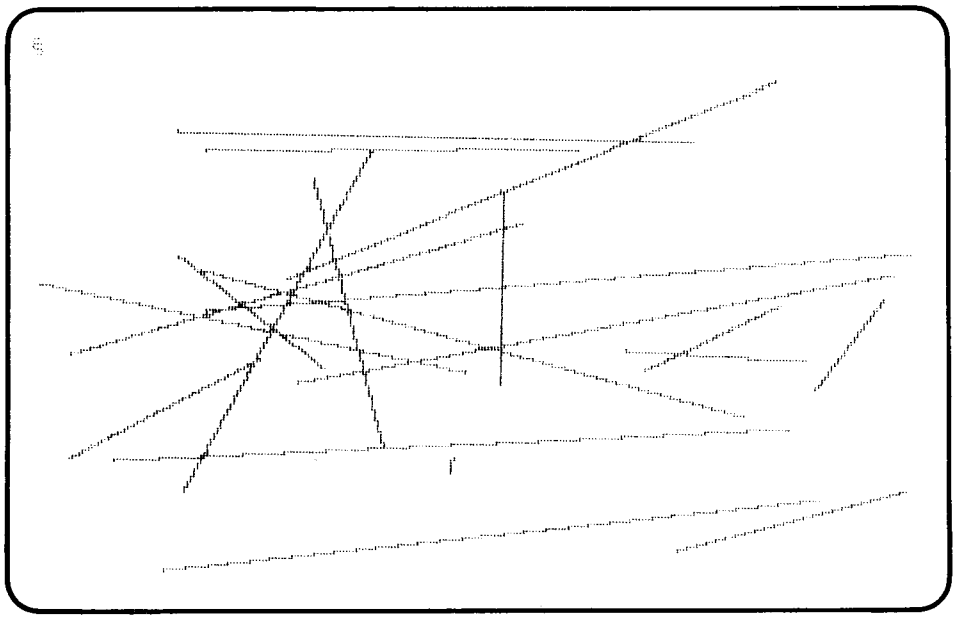
VECTEUR : Tracé d'un vecteur
V1,V2 : Coordonnées du premier point du vecteur à tracer.
V3,V4 : Coordonnées du second point du vecteur à tracer.

VECTEURS ALEATOIRES
V1,V2,V3
V4 : Coordonnées des points des vecteurs à afficher
I : Compteur du nombre de vecteurs à afficher.
R$: Caractère pour fin d'exécution.

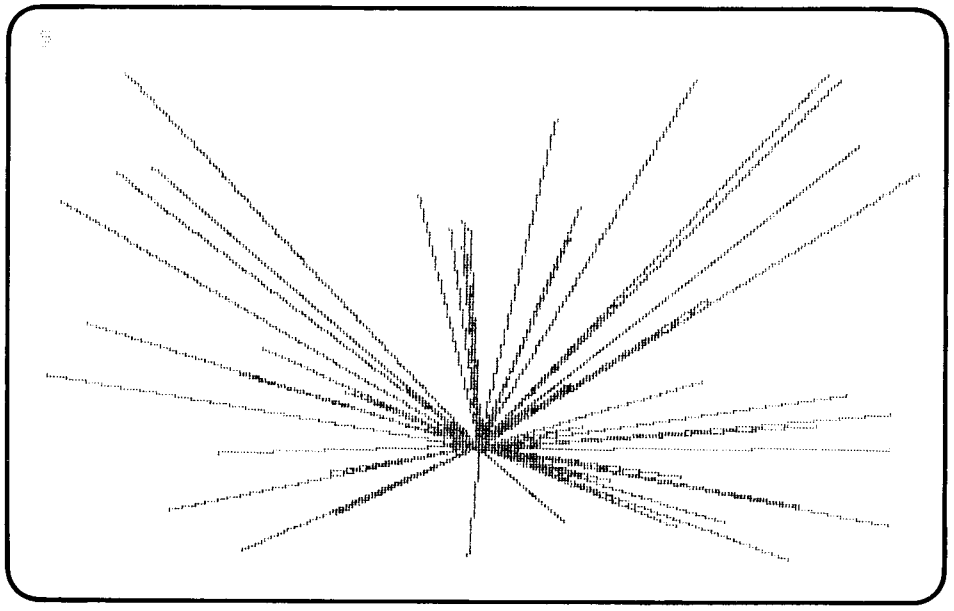
```
1000  REM ------------------------------------------------------------
1002  REM TRACE ALEATOIRE DE VECTEURS 1
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 5000: REM * VECTEURS ALEATOIRES 1 *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM VECTEURS ALEATOIRES 1
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM TRACE ALEATOIRE DE VECTEURS
5020  FOR I = 1 TO 20
5030 V1 =  RND (1) * 279:V2 =  RND (1) * 191
5040 V3 =  RND (1) * 279:V4 =  RND (1) * 191
5050  GOSUB 4000: REM * VECTEUR *
5060  NEXT I
5070  GET R$: IF R$ = "" THEN 5070
5080  HOME : TEXT
5090  RETURN : REM ---------------------------------------------------
5100  REM
```

```
]LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACE ALEATOIRE DE VECTEURS 2
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 5000: REM * VECTEURS ALEATOIRES 2 *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM VECTEURS ALEATOIRES 2
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM TRACE ALEATOIRE DE VECTEURS
5020  FOR I = 1 TO 50
5030 V1 = 140:V2 = 146
5040 V3 =  RND (1) * 279:V4 =  RND (1) * 191
5050  GOSUB 4000: REM * VECTEUR *
5060  NEXT I
5070  GET R$: IF R$ = "" THEN 5070
5080  HOME : TEXT
5090  RETURN : REM ---------------------------------------------------
5100  REM
```

Exemple de tracé aléatoire de vecteurs 1



Exemple de tracé aléatoire de vecteurs 2



VECTEURS PARAMÉTRÉS

Après avoir constaté de visu, que le désordre du hasard peut être esthétique, voyons comment, par une action consciente, nous pouvons faire aussi bien, en organisant les vecteurs. Pour cela, nous allons, dans ce chapitre, substituer le générateur aléatoire par des méthodes consistant à paramétrer les extrémités des vecteurs.

MODELE

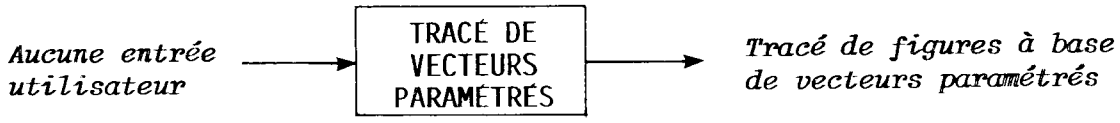
Comme pour les vecteurs aléatoires, nous allons travailler sur les coordonnées des points extrémités des vecteurs à tracer. Mais, cette fois, au lieu de laisser le hasard diriger les opérations, nous allons mettre au point des lois de calculs des coordonnées des vecteurs. Nous ne verrons pas ces lois en détail car elles peuvent être aussi diverses que l'imagination le permet.

Notons seulement qu'il n'est pas nécessaire de posséder un passé mathématique prestigieux pour trouver une loi et que; souvent, une recherche empirique donne de meilleurs résultats et s'avère plus efficace qu'une longue recherche algébrique.

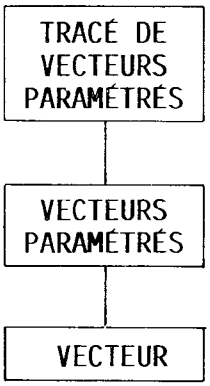
Afin de vous faire une rapide idée des résultats qu'il est possible d'obtenir avec de simples vecteurs, regardez plutôt les exemples réalisés à l'aide des programmes proposés dans ce chapitre.

LE PROGRAMME

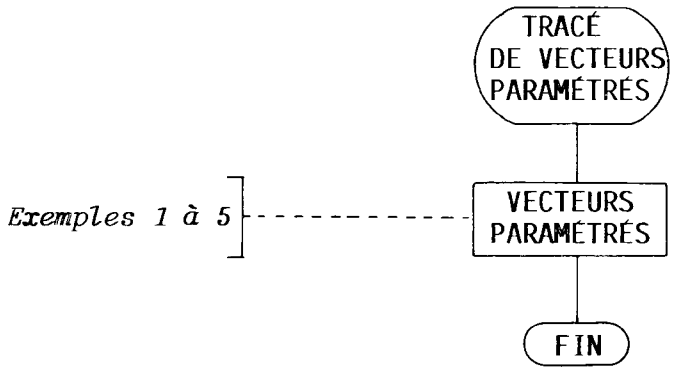
Traitement

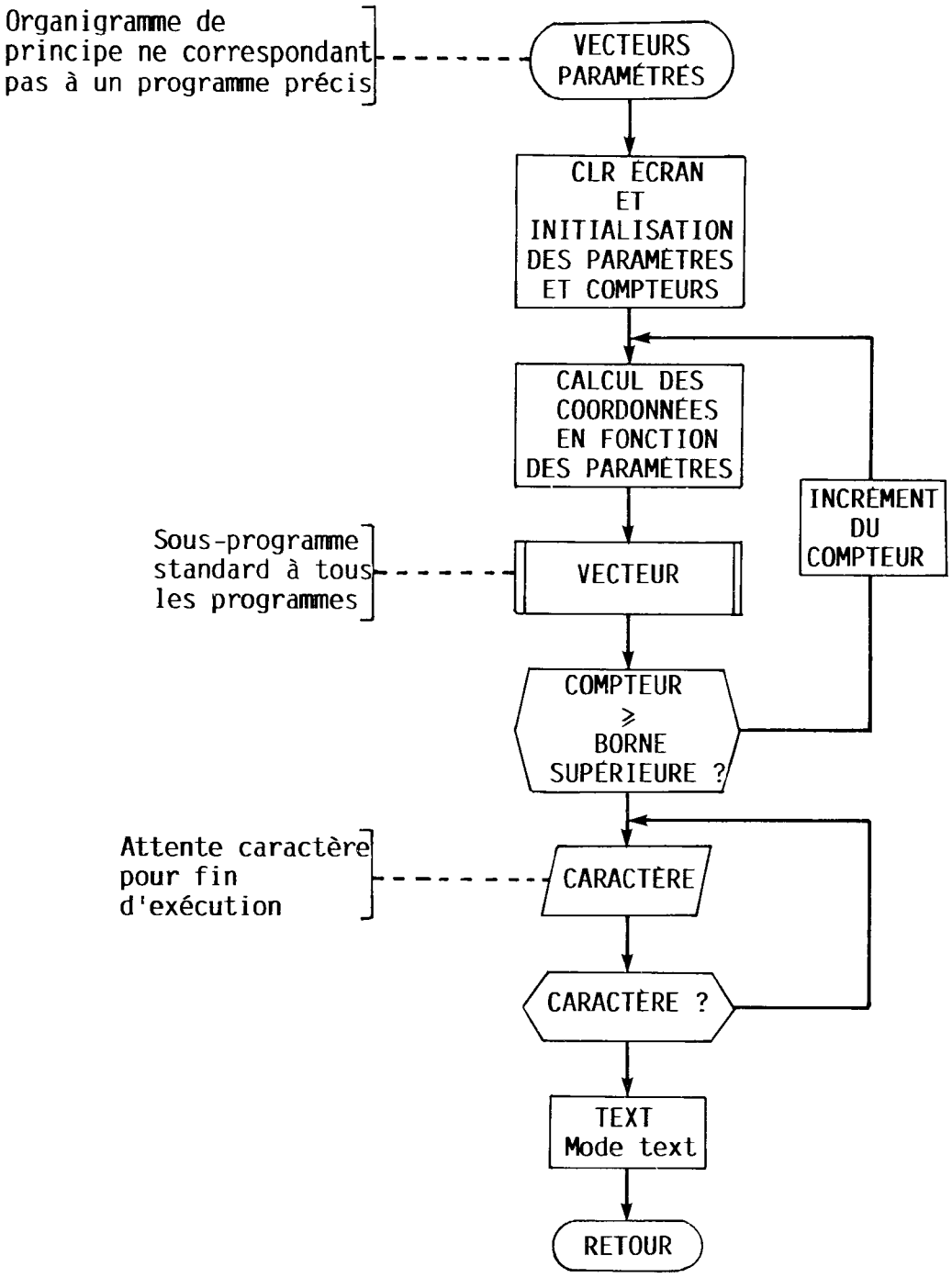


Structure



Organigrammes





Variables

VECTEUR : Tracé d'un vecteur
V1,V2 : Coordonnées du premier point du vecteur à tracer.
V3,V4 : Coordonnées du second point du vecteur à tracer

VECTEURS PARAMETRES 1
01...04,
E1...E4 : Paramètres de construction des vecteurs.
TT : Incrément de changement de position entre chaque vecteur.

R$: Caractère pour fin d'exécution.

VECTEURS PARAMETRES 2

: Voir ci-dessus.

VECTEURS PARAMETRES 3

V1...V4 : Coordonnées des points des vecteurs à tracer.
I : Incrément de changement de position entre chaque
: vecteur et compteur du nombre de vecteurs à tracer.

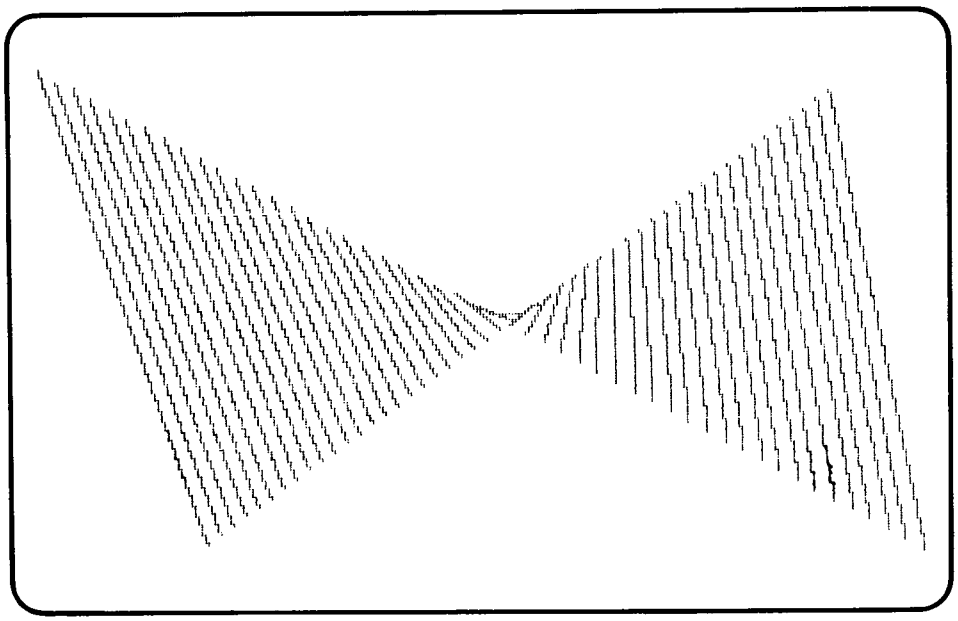
VECTEURS PARAMETRES 4

: Voir ci-dessus.

VECTEURS PARAMETRES 5

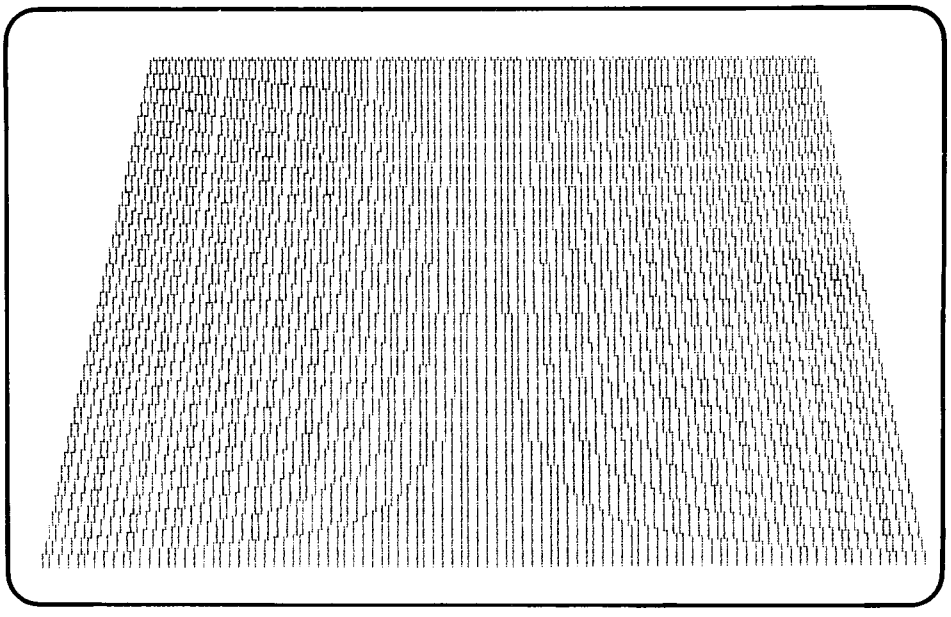
: Voir ci-dessus.

Exemple de tracé de vecteurs parramètrés 1

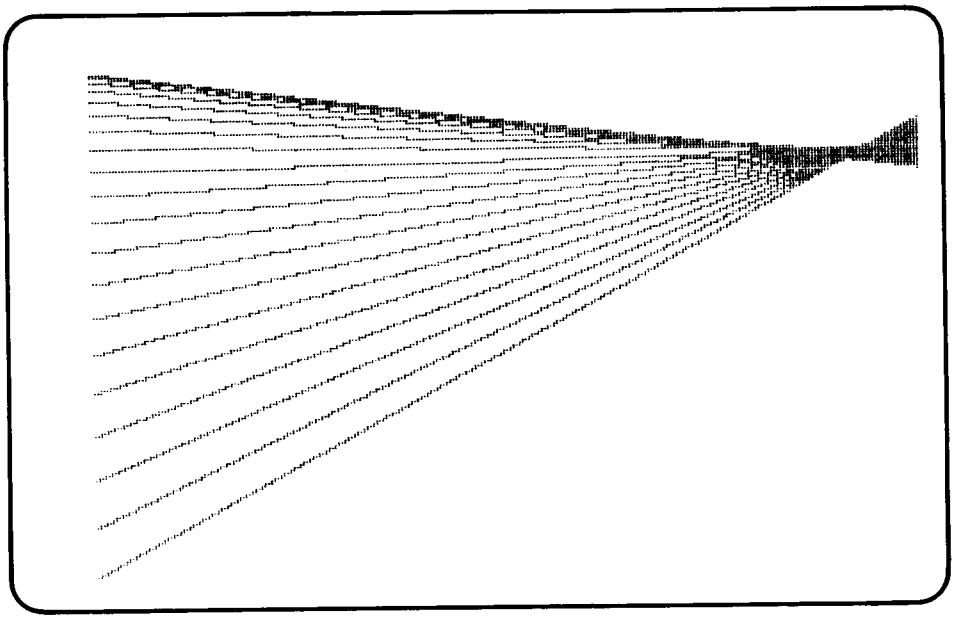


```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACE DE VECTEURS PARAMETRES 1
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 5000: REM * VECTEURS PARAMETRES 1 *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM VECTEURS PARAMETRES 1
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM INITIALISATION DES PARAMETRES
5020 O1 = 0:O2 = 0:E1 = 279:E2 = 191
5030 O3 = 50:O4 = 180:E3 = 250:E4 = 10
5035  REM TRACE
5040  FOR TT = 0 TO 1 STEP .02
5050 V1 = (E1 - O1) * TT + O1
5060 V2 = (E2 - O2) * TT + O2
5070 V3 = (E3 - O3) * TT + O3
5080 V4 = (E4 - O4) * TT + O4
5090  GOSUB 4000: REM * VECTEUR *
5100  NEXT TT
5110  GET R$: IF R$ = "" THEN 5110
5120  HOME : TEXT
5130  RETURN : REM ---------------------------------------------------
5140  REM
```

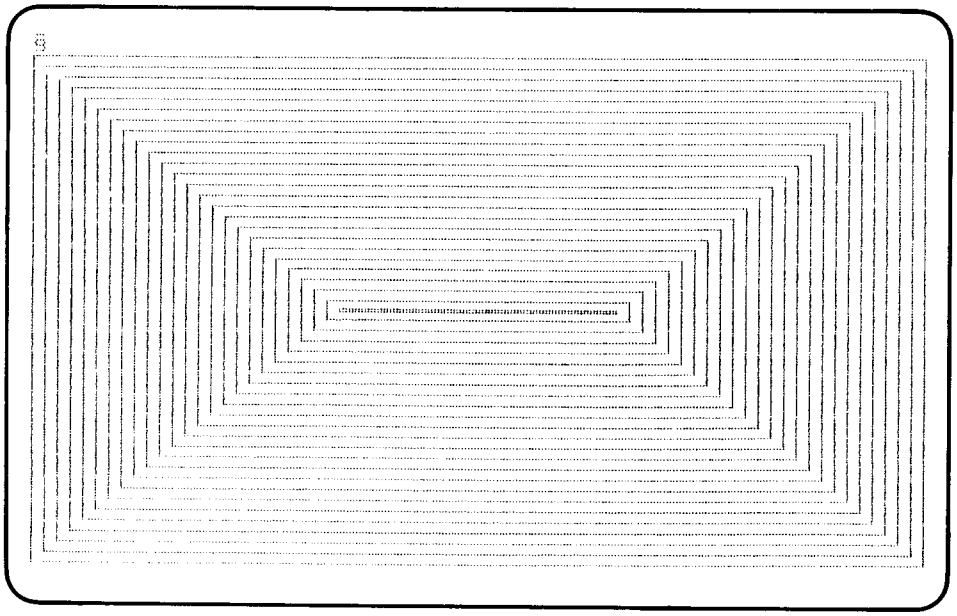
```
]LIST5000-

5000  REM ----------------------------------------
5002  REM VECTEURS PARAMETRES 2
5004  REM ----------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM INITIALISATION DES PARAMETRES
5020 O1 = 0:O2 = 191:E1 = 279:E2 = 191
5030 O3 = 35:O4 = 0:E3 = 244:E4 = 0
5035  REM TRACE
5040  FOR TT = 0 TO 1 STEP .01
5050 V1 = (E1 - O1) * TT + O1
5060 V2 = (E2 - O2) * TT + O2
5070 V3 = (E3 - O3) * TT + O3
5080 V4 = (E4 - O4) * TT + O4
5090  GOSUB 4000: REM * VECTEUR *
5100  NEXT TT
5110  GET R$: IF R$ = "" THEN 5110
5120  HOME : TEXT
5130  RETURN : REM ----------------------------------
5140  REM
```



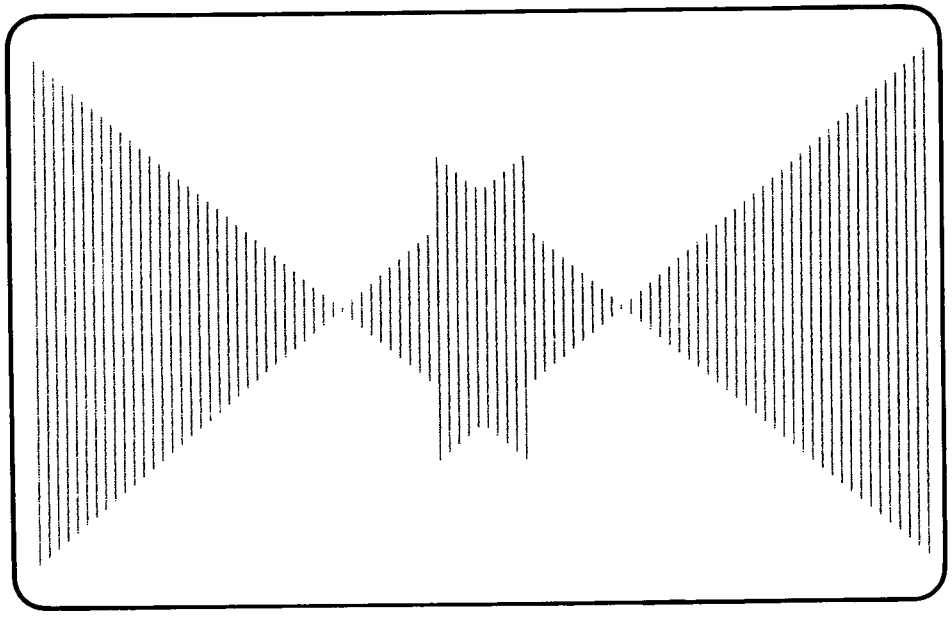
```
$LIST5000-

5000  REM ------------------------------------------------------------
5002  REM VECTEURS PARAMETRES 3
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM TRACE DES VECTEURS
5020  FOR I = 0 TO 19
5030 V1 = 20:V2 = V2 + I
5040 V3 = 279:V4 = 40 - I
5050  GOSUB 4000: REM * VECTEUR *
5060  NEXT I
5070  GET R$: IF R$ = "" THEN 5070
5080  HOME : TEXT
5090  RETURN : REM ------------------------------------------------------
5100  REM
```



```
]LIST5000-

5000  REM ------------------------------------------------------------
5002  REM VECTEURS PARAMETRES 4
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM VECTEURS HORIZONTAUX
5020  FOR I = 0 TO 96 STEP 4
5030 V1 = I:V2 = I:V3 = 279 - I:V4 = I
5040  GOSUB 4000: REM * VECTEUR *
5050 V1 = I:V2 = 191 - I:V3 = 279 - I:V4 = V2
5060  GOSUB 4000: REM * VECTEUR *
5070  NEXT I
5075  REM VECTEURS VERTICAUX
5080  FOR I = 0 TO 92 STEP 4
5090 V1 = I:V2 = I:V3 = I:V4 = 191 - I
5100  GOSUB 4000: REM * VECTEUR *
5110 V1 = 279 - I:V2 = I:V3 = V1:V4 = 191 - I
5120  GOSUB 4000: REM * VECTEUR *
5130  NEXT I
5140  GET R$: IF R$ = "" THEN 5140
5150  HOME : TEXT
5160  RETURN : REM ------------------------------------------------------
5170  REM
```



```
$LIST5000-

5000  REM ----------------------------------------------------------------
5002  REM VECTEURS PARAMETRES 5
5004  REM ----------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM TRACE DES VECTEURS
5020  FOR I = 0 TO 155 STEP 3
5030 V1 = I:V2 = I:V3 = I:V4 = 191 - I
5040  GOSUB 4000: REM * VECTEUR *
5050 V1 = 279 - I:V2 = I:V3 = V1:V4 = 191 - I
5060  GOSUB 4000: REM * VECTEUR *
5070  NEXT I
5080  GET R$: IF R$ = "" THEN 5080
5090  HOME : TEXT
5100  RETURN : REM ---------------------------------------------------------
5110  REM
```



POLYGONES ALÉATOIRES

Dans ce chapitre, nous allons nous donner les moyens de disposer aléatoirement des polygones à quatre côtés parallèles deux à deux, en un mot : des rectangles. Mais la méthode utilisée pour atteindre ce but, peut être généralisée aux autres représentations graphiques de base. Ainsi, vous serez en mesure de disposer aléatoirement : des triangles, des trapèzes, des ellipses... etc.

MODELE

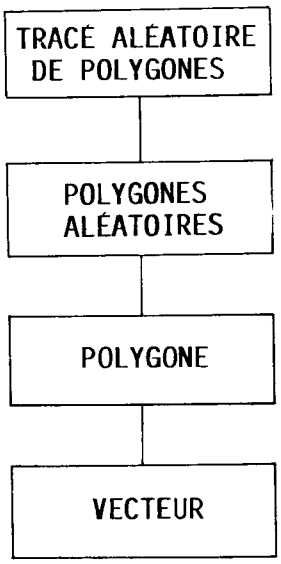
Lors de chaque définition d'un sous-programme permettant de tracer une nouvelle représentation graphique élémentaire, nous avons pris soin d'en faire un bloc autonome auquel il faut fournir les caractéristiques de la figure à tracer en entrée, afin d'obtenir un affichage de la figure souhaitée après exécution. Cette option nous permet, à partir d'ensembles différents de valeurs de caractéristiques, d'obtenir des tracés différents d'une même figure. Aussi, en générant aléatoirement ces ensembles de valeurs dans les limites permises, nous obtiendrons le résultat recherché qui consiste à disposer et à générer des représentations sur l'écran de façon aléatoire.

C'est comme nous venons de le décrire que le programme de génération aléatoire de rectangle opère : **P1, P2, P3, P6** calculés à l'aide du générateur de nombres aléatoires, définissent la longueur et la largeur du rectangle. Ensuite, il suffit d'affecter **P7, P4, P5, P8** qui dépendent des variables calculées auparavant et d'appeler le sous-programme ***POLYGONE*** pour tracer un rectangle qui, à chaque calcul aura une taille et une position différentes sur l'écran.

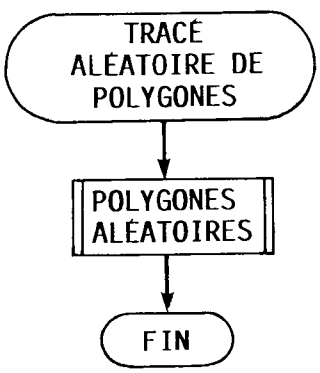
LE PROGRAMME

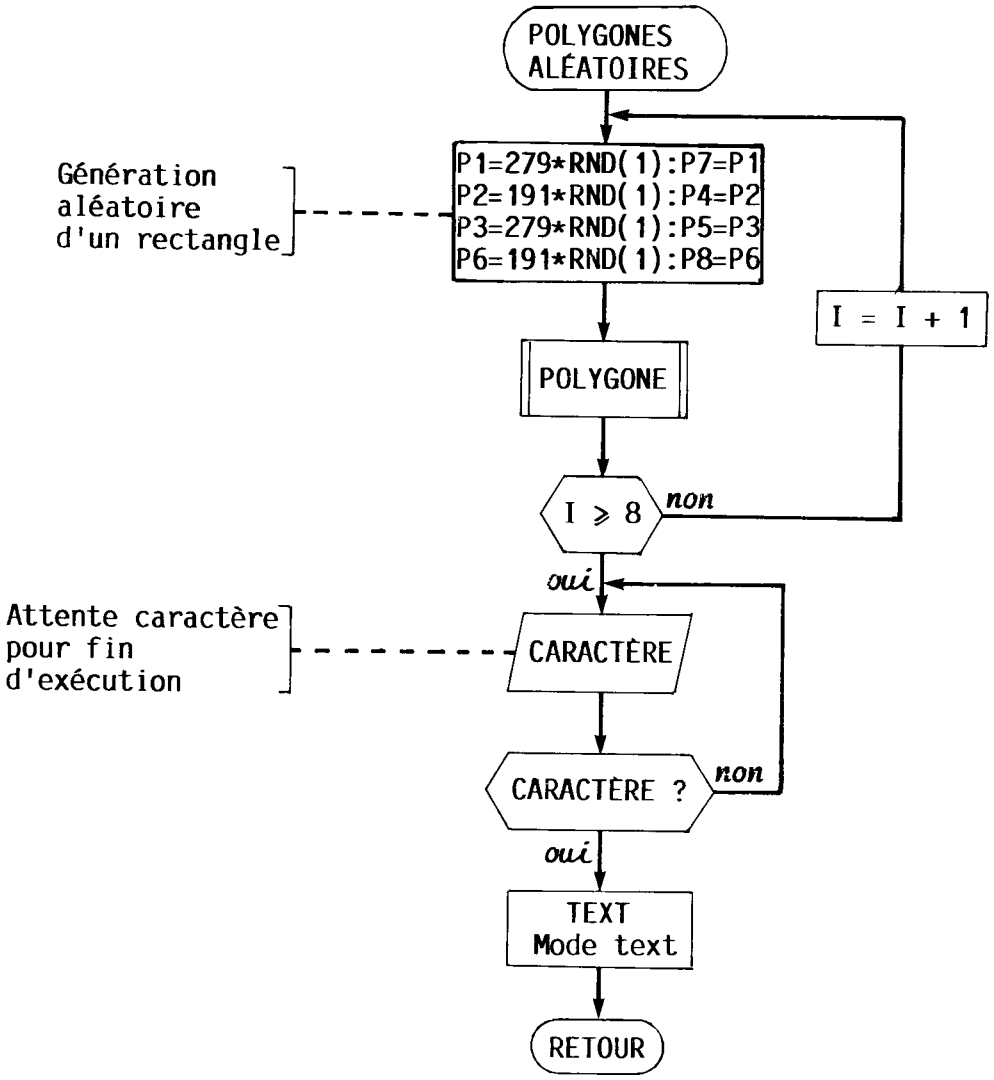
Traitement

Structure



Organigrammes





Variables

VECTEUR : Tracé d'un vecteur

V1,V2 : Coordonnées du premier point du vecteur à tracer.
V3,V4 : Coordonnées du second point du vecteur à tracer.

POLYGONE : Tracé d'un polygone

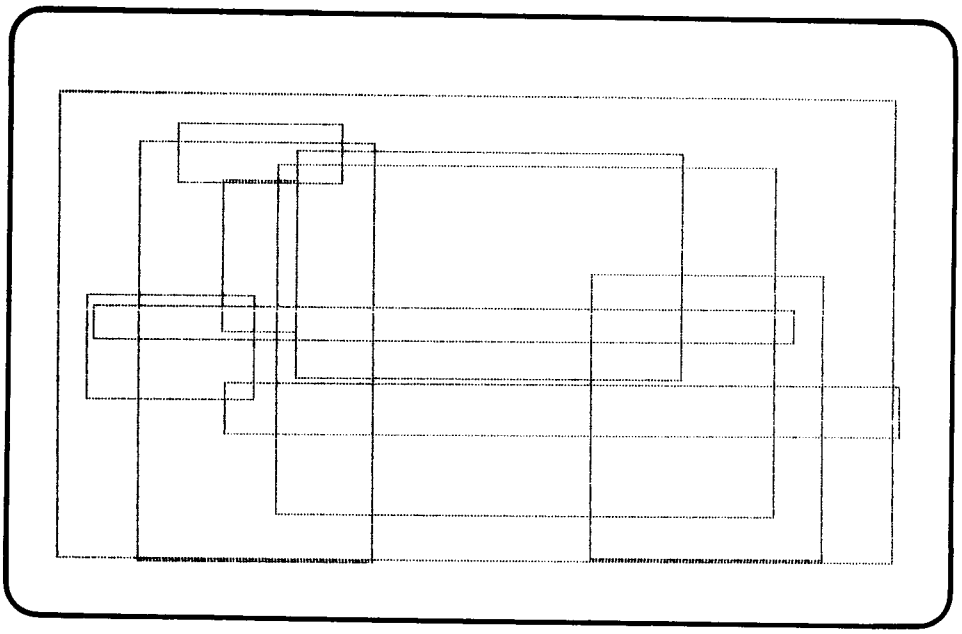
P1...P8 : Coordonnées des points du polygone à tracer.
V1...V4 : Coordonnées des points extrémités des vecteurs à tracer.

POLYGONES ALEATOIRES

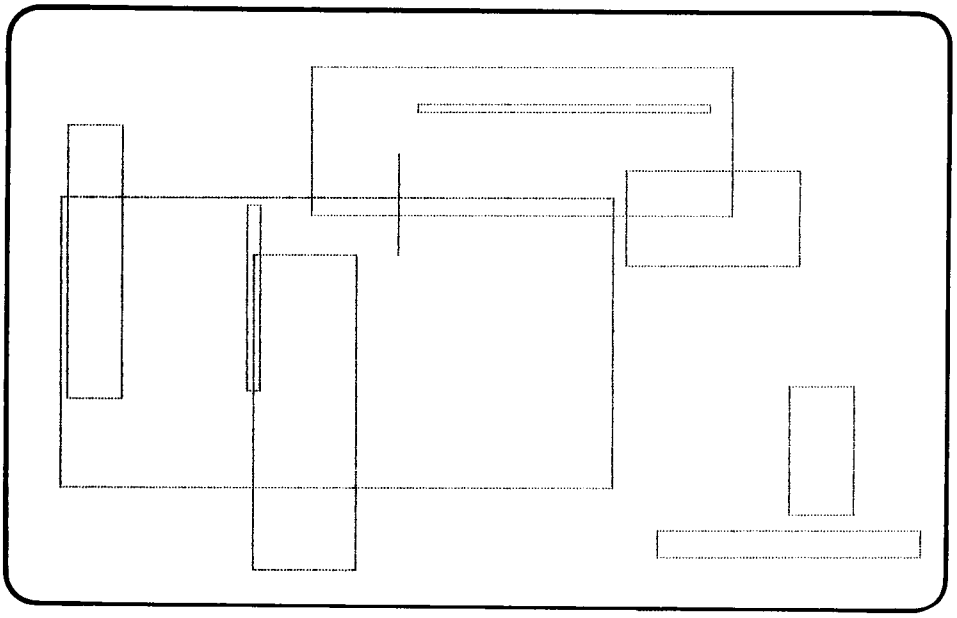
P1...P8 : Coordonnées des points des polygones à tracer.
I : Compteur du nombre de polygones à tracer.
R$: Caractère pour fin d'exécution.

```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACE ALEATOIRE DE POLYGONES
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 6000: REM * POLYGONES ALEATOIRES *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM POLYGONE
5004  REM ------------------------------------------------------------
5005  REM (P1,P2)...(P7,P8) : COUPLES DE COORDONNEES
5006  REM DES POINTS DU POLYGONE A TRACER
5010 V1 = P1:V2 = P2:V3 = P3:V4 = P4
5020  GOSUB 4000: REM *VECTEUR*
5030 V1 = P3:V2 = P4:V3 = P5:V4 = P6
5040  GOSUB 4000: REM * VECTEUR *
5050 V1 = P5:V2 = P6:V3 = P7:V4 = P8
5060  GOSUB 4000: REM * VECTEUR *
5070 V1 = P7:V2 = P8:V3 = P1:V4 = P2
5080  GOSUB 4000: REM * VECTEUR *
5090  RETURN : REM ---------------------------------------------------
5100  REM
5110  REM
6000  REM ------------------------------------------------------------
6002  REM POLYGONES ALEATOIRES
6004  REM ------------------------------------------------------------
6005  REM PAGE 1 GRAPHIQUE
6010  HGR : POKE  - 16302,0: HCOLOR= 3
6015  REM TRACE DES POLYGONES ALEATOIRES
6020  FOR I = 1 TO 10
6030 P1 = 279 *  RND (1):P7 = P1
6040 P2 = 191 *  RND (1):P4 = P2
6050 P3 = 279 *  RND (1):P5 = P3
6060 P6 = 191 *  RND (1):P8 = P6
6070  GOSUB 5000: REM * POLYGONE *
6080  NEXT I
6090  GET R$: IF R$ = "" THEN 6090
6100  HOME : TEXT
6110  RETURN : REM ---------------------------------------------------
6120  REM
```

Exemple de tracé aléatoire de polygones 1



Exemple de tracé aléatoire de polygones 2



Deuxième partie

EXPRESSION GRAPHIQUE MATHÉMATIQUE

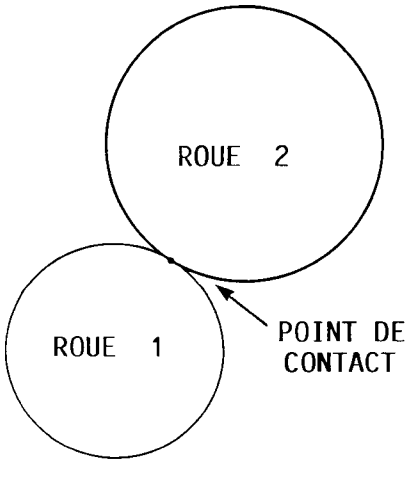
Représentation graphique d'équations

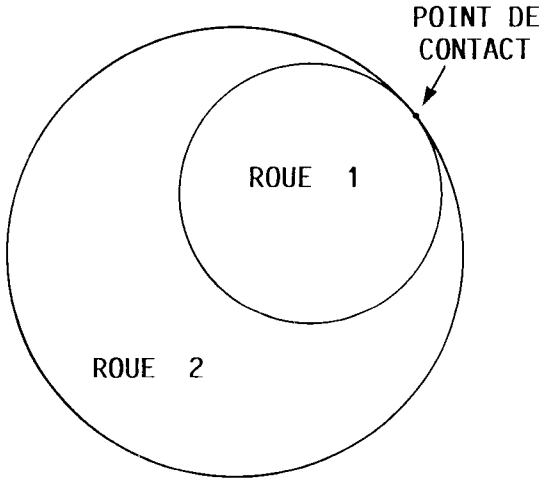
HYPO ET ÉPI-CYCLOÏDES

Si le nom de ce chapitre peut paraître barbare, il ne faut pas pour autant, passer au chapitre suivant car il cache de très esthétiques représentations graphiques aussi variées que simples à réaliser.

HYPOTHESE

Depuis l'invention de la roue, les mathématiques ont progressé considérablement et, les mathématiciens les plus "matheux" ont eu l'idée d'associer deux roues. Mais ce n'est pas la charrette qu'ils venaient d'inventer, car les hasards de la science ne l'avaient pas voulu ainsi. Les deux roues qu'ils venaient d'associer se touchaient par un point et un seul, n'avaient pas forcément le même rayon et se trouvaient toutes deux dans un même plan. Pratiquement, après de très longues journées de réflexion, ils aboutirent aux deux schémas suivants :





Alors leur vint une nouvelle idée. Ils étudièrent le mouvement d'un point de la roue 1 lorsque celle-ci roule sans glisser à l'extérieur, ou encore à l'intérieur de la roue 2.

Dans le premier cas de figure, ils appelèrent la courbe **épi-cycloïde** et montrèrent qu'elle avait pour équation :

$$X = (U+V)\cos(T) + V.\sin((U+V)\,T / V)$$
$$Y = (U+V)\sin(T) + V.\cos((U+V)\,T / V)$$

Dans le second cas, ils appelèrent la courbe **hypo-cycloïde** et montrèrent qu'elle avait l'équation suivante :

$$X = (U-V)\cos(T) + V\cos((U-V)\,T / V)$$
$$Y = (U-V)\sin(T) - V\sin((U-V)\,T / V)$$

Sans démontrer ces équations puisque tel n'est pas notre propos, nous allons les utiliser pour tracer les courbes représentant le déplacement du point fixé sur la roue 1.

MODELE

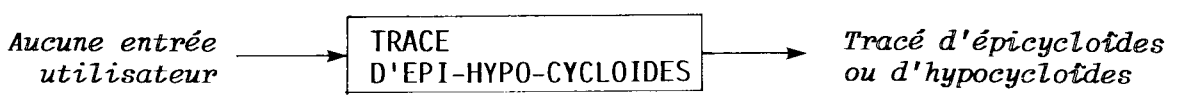
Après avoir fixé les variables **U** et **V**, nous disposons de deux équations fonctions de l'angle **T**. Alors, pour tracer les courbes, nous effectuons un changement de repère en utilisant les variables **01** et **02**, puis nous incrémentons l'angle **T** de zéro à une valeur suffisante pour que la courbe se referme sur elle-même. A chaque calcul des coordonnées **X** et **Y**, l'utilisation du sous-programme ***POINT*** permet le tracé point par point, de la courbe.

EXEMPLE D'UTILISATION

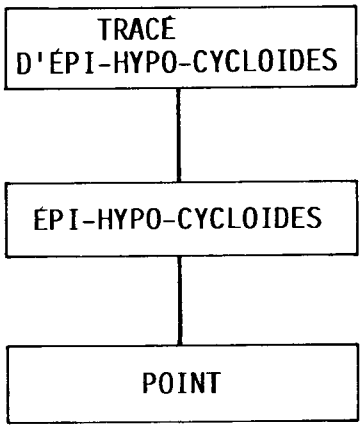
Faites comme nous l'avons fait aux pages 80 à 83, modifiez les valeurs des variables **U** et **V** puis cadrez les représentations à l'aide des variables 01 et 02, vous obtiendrez ainsi une grande variété de courbes planes toutes plus belles les unes que les autres.

LE PROGRAMME

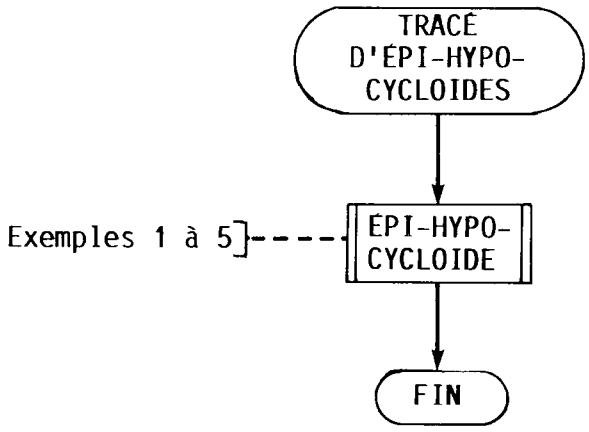
Traitement

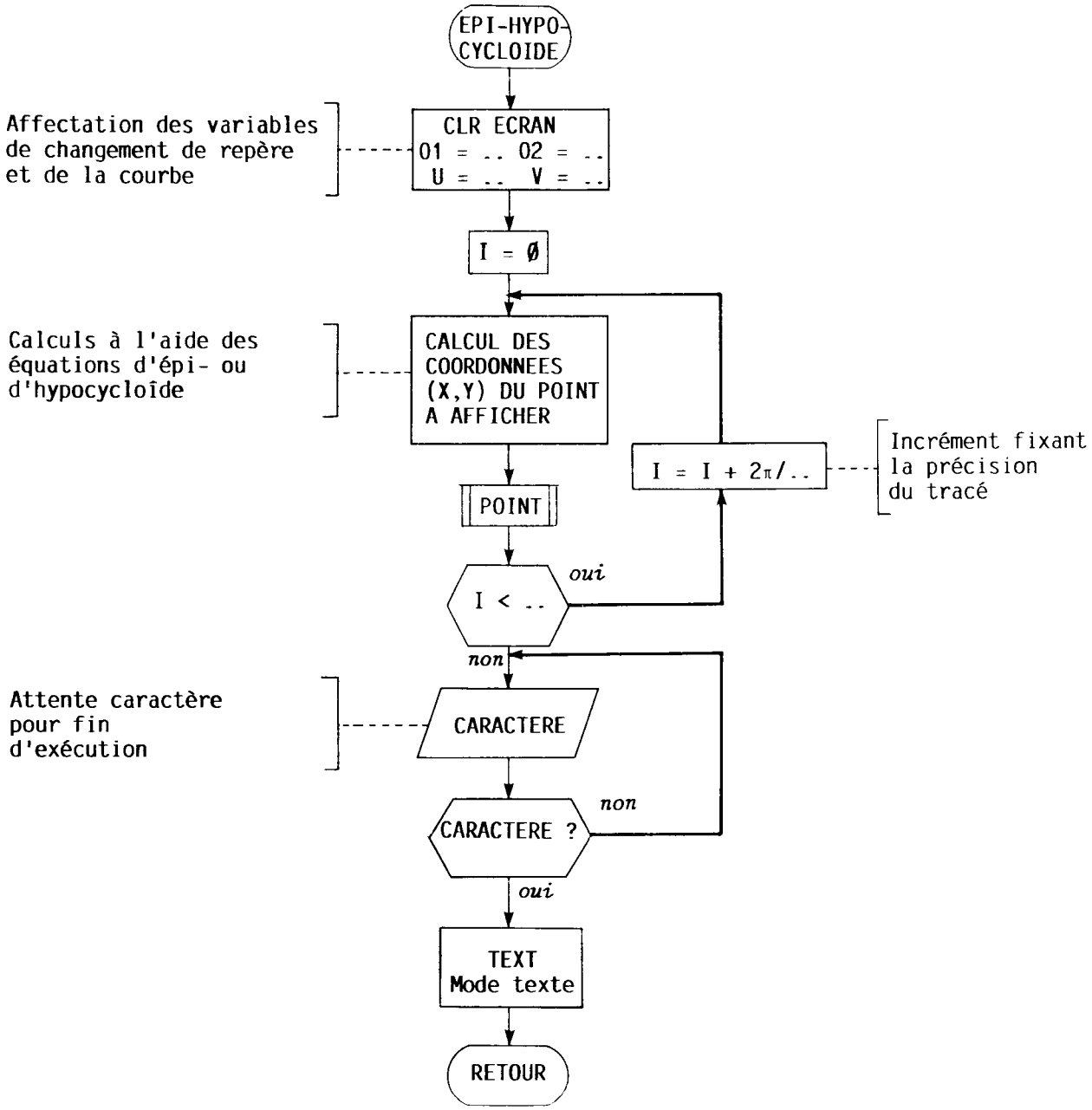


Structure



Organigrammes





Variables

POINT : Affichage d'un point
X : Abscisse du point à afficher
Y : Coordonnée du point à afficher

EPI-HYPO-CYCLOIDES : Tracé d'une courbe représentative

01,02 : Changement d'origine du nouveau repère par rapport à l'ancien
U,V : Paramètres de l'hypo ou de l'épicycloïde

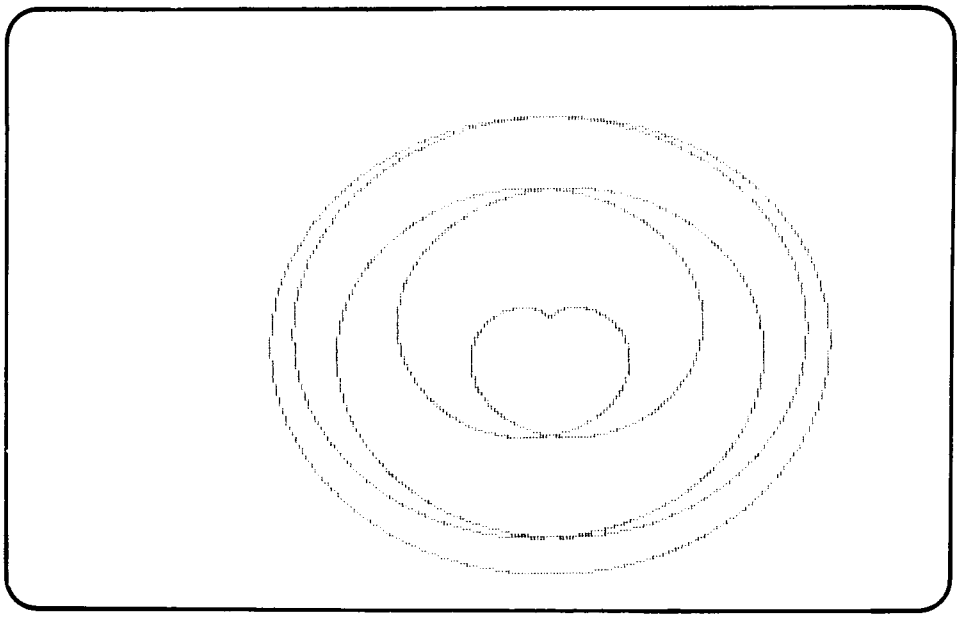
I	Angle en radian entre l'axe des abscisses et la droite passant par l'origine et le point à afficher
R$	Variable test de fin d'exécution

Particularités

Nous remarquerons les modifications apportées aux équations du dernier sous-programme qui ne sont plus celles d'une hypocycloïde bien qu'elles en soient très proches et que la représentation tend à faire penser le contraire.

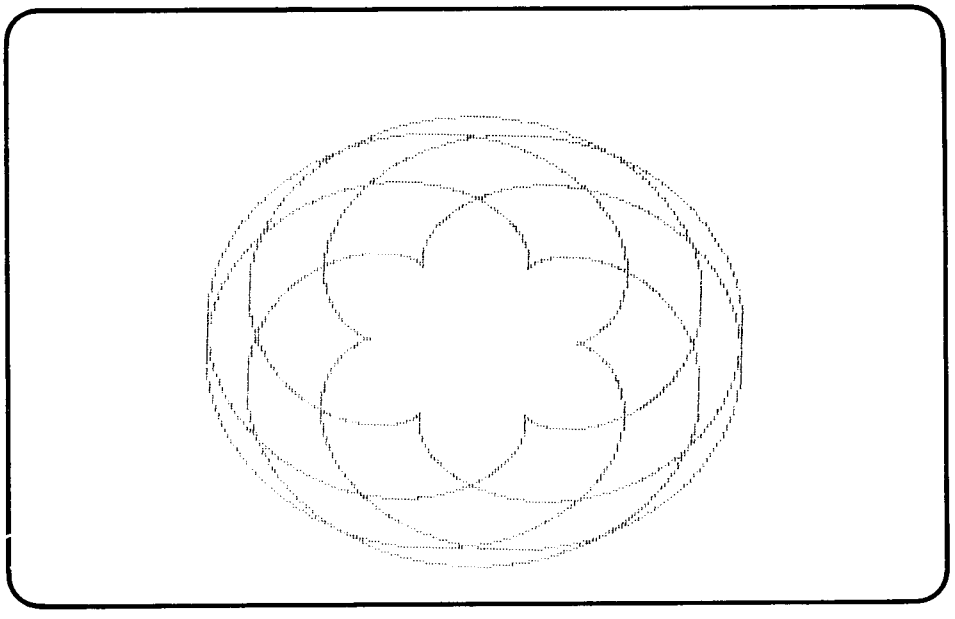
Seul un programme est listé dans son intégralité et, pour obtenir le tracé de l'ensemble des exemples d'exécution, il suffit d'en modifier le sous-programme EPI-HYPO-CYCLOIDE comme l'indiquent les routines listées.

Exemple d'épi-hypo-cycloïdes 1

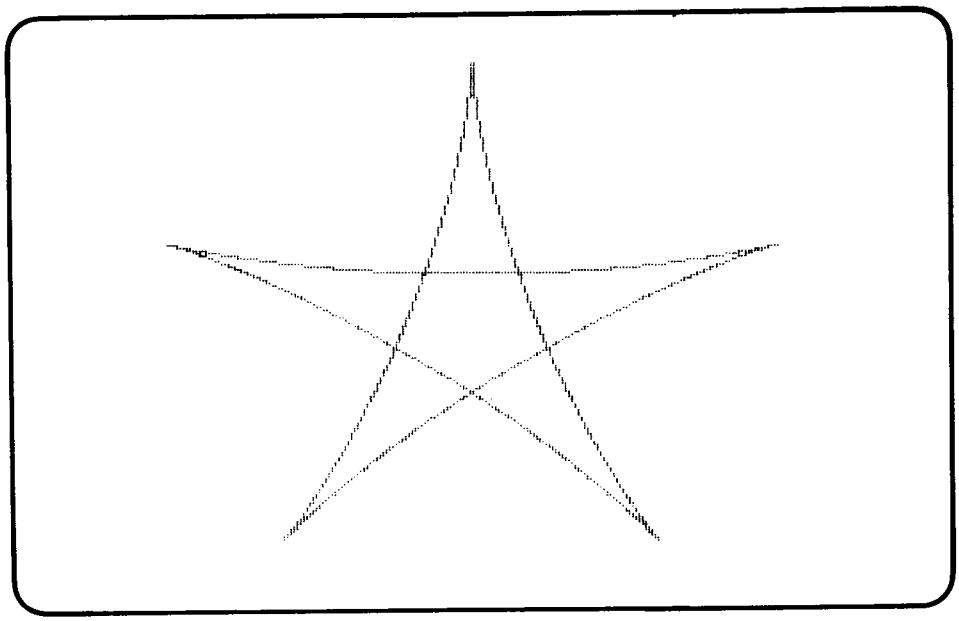


```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACE D'EPI-HYPO-CYCLOIDES 1
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 4000: REM * EPI-HYPO-CYCLOIDE 1 *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
3000  REM ------------------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ---------------------------------------------------
3070  REM
3080  REM
4000  REM ------------------------------------------------------------
4002  REM EPI-HYPO-CYCLOIDE 1
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE + TEXTE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM COORDONNEES DE L'ORIGINE
4020 O1 = 146:O2 = 96
4025  REM PARAMETRES DE L'EPI-HYPO-CYCLOIDE
4030 U = 8:V = 40
4035  REM TRACE DE L'EPI-HYPO-CYCLOIDE
4040  FOR I = .01 TO 32 STEP .01
4050 X = O1 + (U + V) *  SIN (I) + V *  SIN ((U + V) * I
     / V)
4060 Y = O2 + (U + V) *  COS (I) + V *  COS ((U + V) * I
     / V)
4070  GOSUB 3000: REM * POINT *
4080  NEXT I
4090  GET R$: IF R$ = "" THEN 4090
4100  HOME : TEXT
4110  RETURN : REM ---------------------------------------------------
4120  REM
```

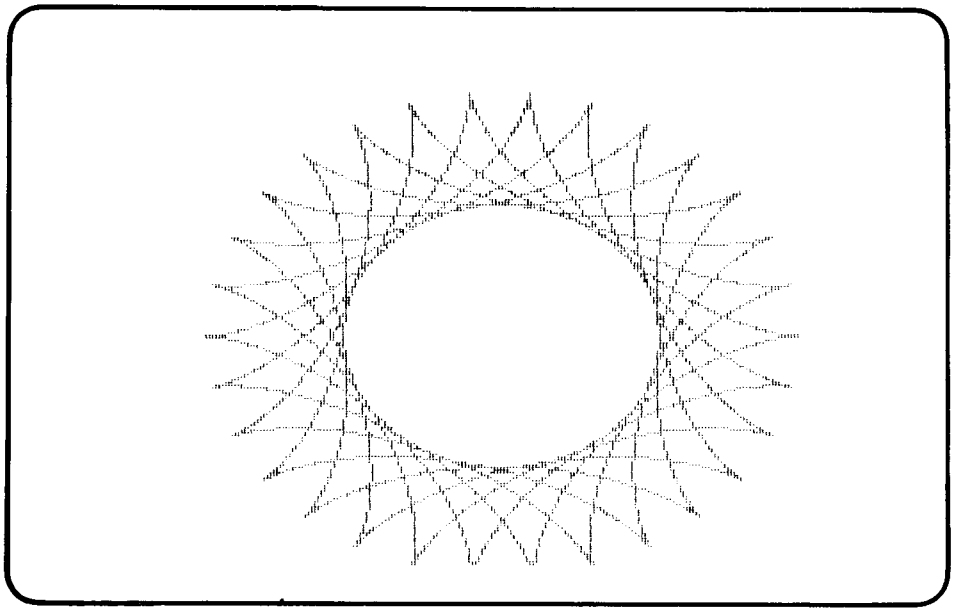
```
$LIST4000-

4000  REM ------------------------------------------------------------
4002  REM EPI-HYPO-CYCLOIDE 2
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE + TEXTE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM COORDONNEES DE L'ORIGINE
4020 O1 = 146:O2 = 96
4025  REM PARAMETRES DE L'EPI-HYPO-CYCLOIDE
4030 U = 32:V = 80 / 3
4035  REM TRACE DE L'EPI-HYPO-CYCLOIDE
4040  FOR I = .009 TO 32 STEP .009
4050 X = O1 + (U + V) *  SIN (I) + V *  SIN ((U + V) * I
     / V)
4060 Y = O2 + (U + V) *  COS (I) + V *  COS ((U + V) * I
     / V)
4070  GOSUB 3000: REM * POINT *
4080  NEXT I
4090  GET R$: IF R$ = "" THEN 4090
4100  HOME : TEXT
4110  RETURN : REM ------------------------------------------------------
4120  REM
```



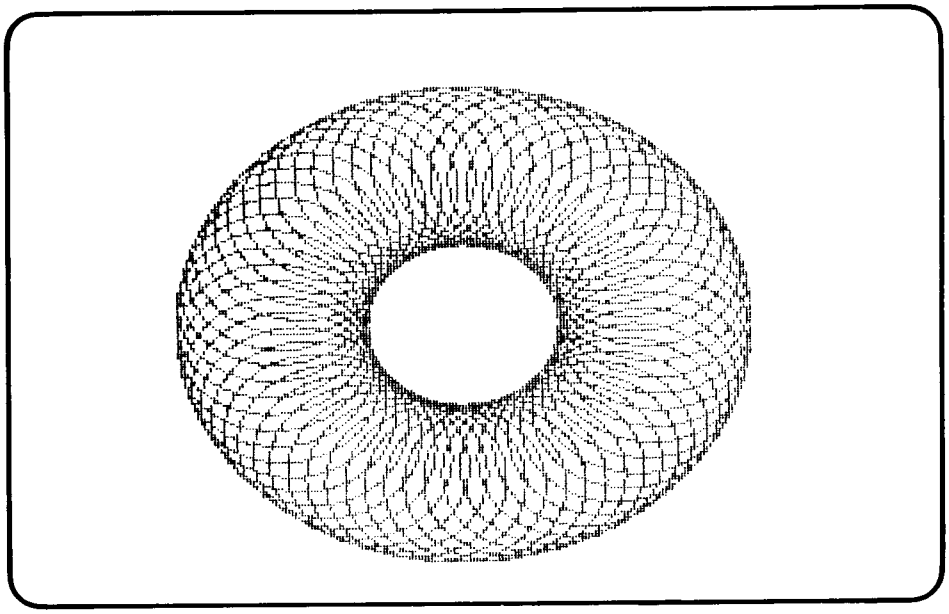
```
$LIST4000-

4000  REM ------------------------------------------------------------
4002  REM EPI-HYPO-CYCLOIDE 3
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE + TEXTE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM COORDONNEES DE L'ORIGINE
4020 O1 = 146:O2 = 105
4025  REM PARAMETRES DE L'EPI-HYPO-CYCLOIDE
4030 U = 100:V = 60
4035  REM TRACE DE L'EPI-HYPO-CYCLOIDE
4040  FOR I = .01 TO 19.5 STEP .01
4050 X = O1 + (U - V) *  SIN (I) + V *  SIN ((U - V) * I
     / V)
4060 Y = O2 + (U - V) *  COS (I) - V *  COS ((U - V) * I
     / V)
4070  GOSUB 3000: REM * POINT *
4080  NEXT I
4090  GET R$: IF R$ = "" THEN 4090
4100  HOME : TEXT
4110  RETURN : REM ------------------------------------------------------
4120  REM
```



```
§LIST4000-

4000  REM ----------------------------------------------------------------
4002  REM EPI-HYPO-CYCLOIDE 4
4004  REM ----------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE + TEXTE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM COORDONNEES DE L'ORIGINE
4020 O1 = 146:O2 = 105
4025  REM PARAMETRES DE L'EPI-HYPO-CYCLOIDE
4030 U = 120 / 1.3:V = 28 / 1.3
4035  REM TRACE DE L'EPI-HYPO-CYCLOIDE
4040  FOR I = .007 TO 44 STEP .007
4050 X = O1 + (U - V) *  SIN (I) + V *  SIN ((U - V) * I
     / V)
4060 Y = O2 + (U - V) *  COS (I) - V *  COS ((U - V) * I
     / V)
4070  GOSUB 3000: REM * POINT *
4080  NEXT I
4090  GET R$: IF R$ = "" THEN 4090
4100  HOME : TEXT
4110  RETURN : REM ----------------------------------------------------------
4120  REM
```



```
$LIST4000-

4000  REM ------------------------------------------------------------
4002  REM EPI-HYPO-CYCLOIDE 5
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE + TEXTE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM COORDONNEES DE L'ORIGINE
4020 O1 = 146:O2 = 95
4025  REM PARAMETRES DE L'EPI-HYPO-CYCLOIDE
4030 U = 90:V = 30
4035  REM TRACE DE L'EPI-HYPO-CYCLOIDE
4040  FOR I = .0005 TO 6.5 STEP .0005
4050 X = O1 + (U - V) *  SIN (I) + V *  SIN ((U - V) * I
    )
4060 Y = O2 + (U - V) *  COS (I) - V *  COS ((U - V) * I
    )
4070  GOSUB 3000: REM * POINT *
4080  NEXT I
4090  GET R$: IF R$ = "" THEN 4090
4100  HOME : TEXT
4110  RETURN : REM ------------------------------------------------------
4120  REM
```



FONCTIONS CIRCULAIRES

Dans ce chapitre, nous allons envisager un certain nombre d'équations dont le point commun est qu'elles sont toutes paramétrées en fonction d'un angle. Nous pourrons alors constater à quel point est grand le nombre de représentations graphiques réalisables simplement.

MODELE

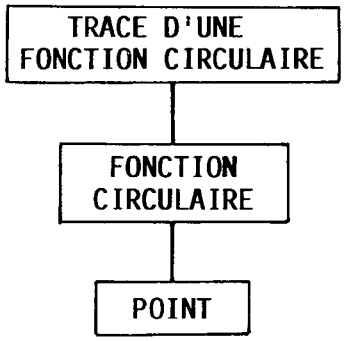
Nous allons voir un ensemble de sous-programme d'origine 4000, réalisant les représentations graphiques d'équations fonctions d'un angle. Chaque sous-programme est composé d'une équation qui permet le calcul des coordonnées du point à afficher. Parfois, comme dans le cas de ***cycloïdes***, la même courbe est représentée plusieurs fois afin d'obtenir un effet supplémentaire mais le principe du calcul des coordonnées reste le même.

Les équations choisies donnent des effets spectaculaires. Pour en juger, il suffit de consulter les pages qui suivent, mais sans aucun doute, de nombreux autres calculs sont possibles avec des représentations toujours plus originales. Il ne reste plus qu'à essayer.

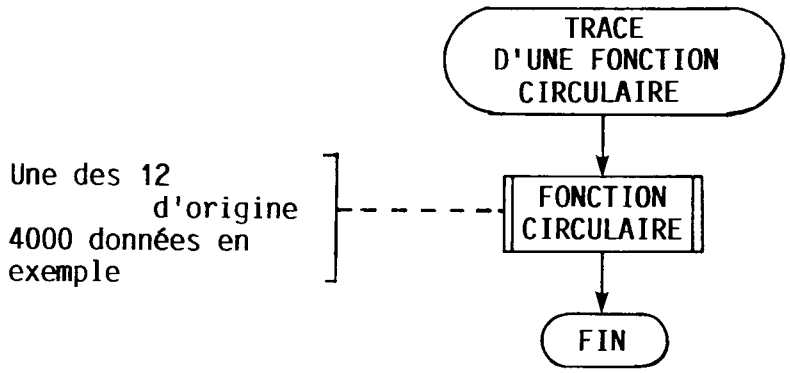
LE PROGRAMME

Traitement

Structure



Organigramme



Variables

POINT : Affichage d'un point

X	Abscisse du point à afficher
Y	Ordonnée du point à afficher

FONCTION CIRCULAIRE

TB,TH	Limites inférieure et supérieure de l'angle de rotation
01,02	Coordonnée de la nouvelle origine
PR	Précision du tracé ou incrément de calcul des points à afficher
I	Indice de répétition du tracé
T	Angle de rotation du tracé
R	Rayon du tracé
U,V	Paramètres de la cycloïde
R$	Variable pour le caractère de fin d'exécution

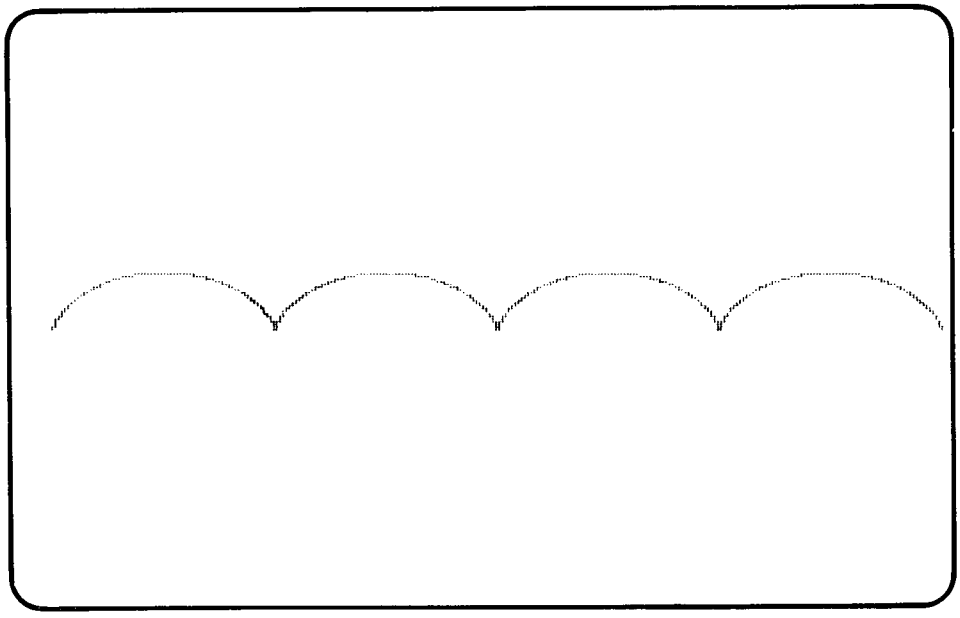
Particularités

- Les sous-programmes dont les numéros de ligne commencent par l'adresse 4000 peuvent tous être exécutés en étant mis à la place du sous-programme ***CYCLOIDE*** dans la première liste.

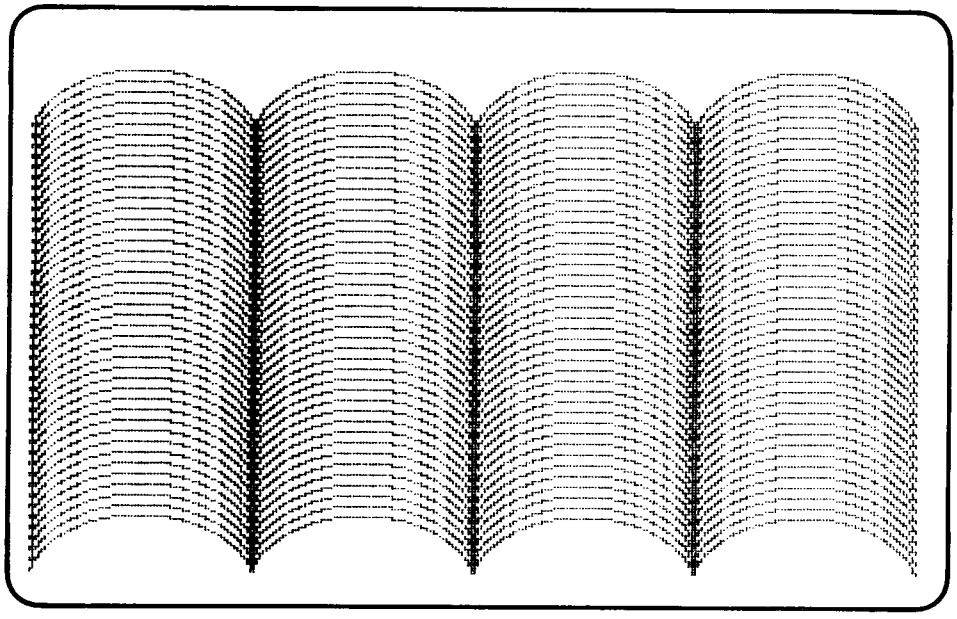
- Le sous-programme ***CYCLOIDE*** est listé afin de montrer comment il est possible de rendre des effets graphiques en juxtaposant simplement des figures simples comme dans le sous-programme ***CYCLOIDES.***

```
§LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACE D'UNE FONCTION CIRCULAIRE 1
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 4000: REM * CYCLOIDE *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
3000  REM ------------------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ---------------------------------------------------
3070  REM
3080  REM
```

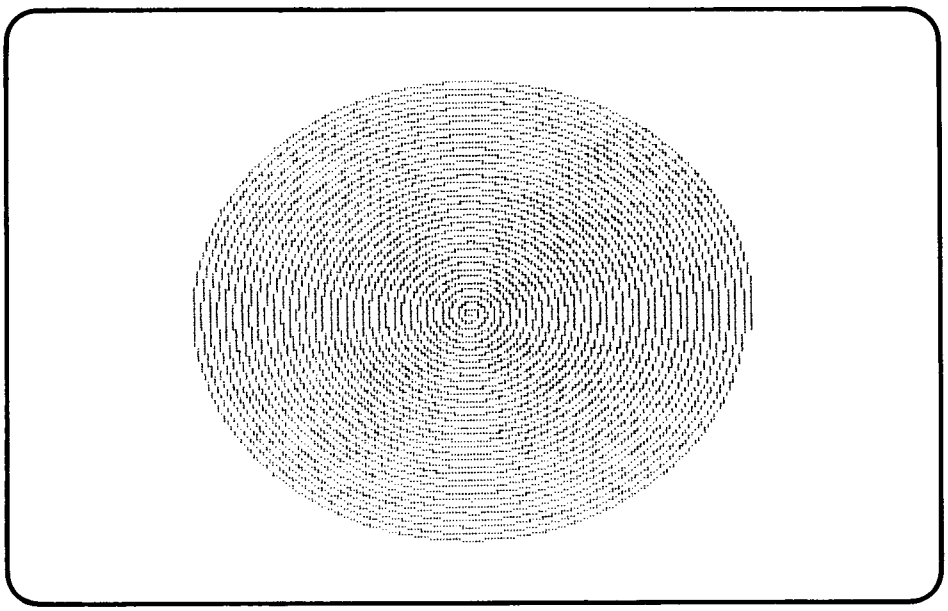
```
4000  REM ------------------------------------------------------------
4002  REM CYCLOIDE
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM COORDONNEES DE L'ORIGINE
4020 O1 = 140:O2 = 95
4025  REM PARAMETRES DE LA CYCLOIDE
4030 U =  - 11
4035  REM TRACE DES CYCLOIDES
4040  FOR I =  - 12.56 TO 12.56 STEP .03
4050 X = O1 + U * (I -  SIN (I))
4060 Y = O2 + U * (1 -  COS (I))
4070  GOSUB 3000: REM * POINT *
4080  NEXT I
4090  GET R$: IF R$ = "" THEN 4090
4100  HOME : TEXT
4110  RETURN : REM ------------------------------------------------------
4120  REM
```



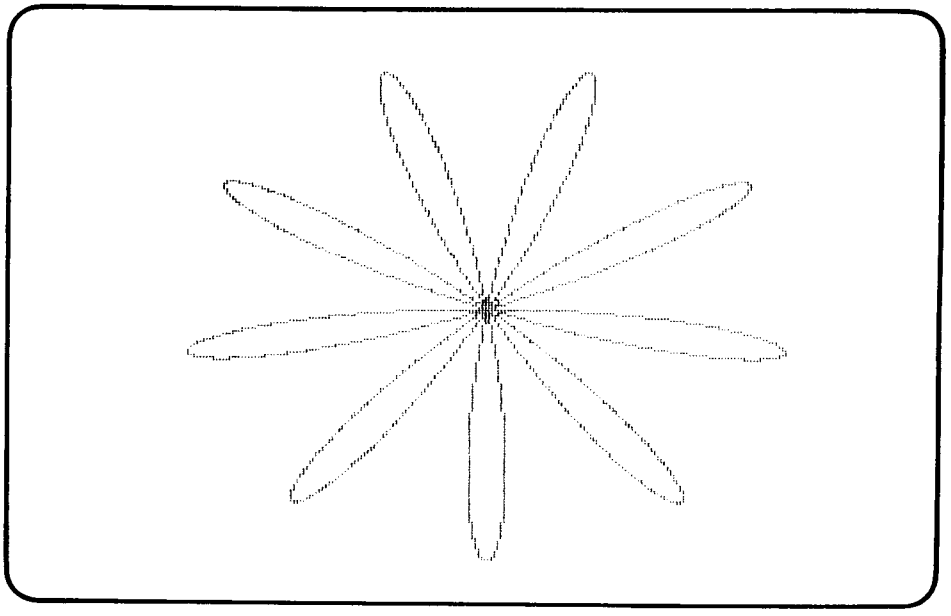
```
$LIST4000-

4000  REM ------------------------------------------------------------
4002  REM CYCLOIDES
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM COORDONNEES DE L'ORIGINE
4020 O1 = 140:O2 = 95
4025  REM PARAMETRES DE LA CYCLOIDE
4030 U =  - 11
4035  REM TRACE DES CYCLOIDES
4040  FOR O2 = 24 TO 192 STEP 4
4050  FOR I =  - 12.56 TO 12.56 STEP .03
4060 X = O1 + U * (I -  SIN (I))
4070 Y = O2 + U * (1 -  COS (I))
4080  GOSUB 3000: REM  * POINT *
4090  NEXT I
4100  NEXT O2
4110  GET R$: IF R$ = "" THEN 4110
4120  HOME : TEXT
4130  RETURN : REM ------------------------------------------------------
4140  REM
```



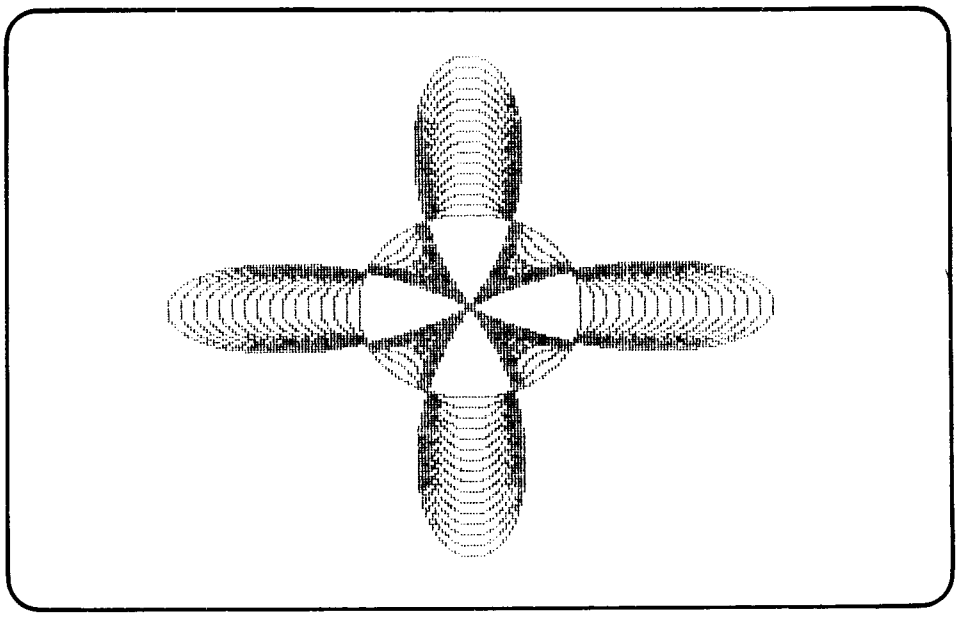
```
]LIST4000-

4000  REM ----------------------------------------
4002  REM SPIRALE
4004  REM ----------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM COORDONNEES DE L'ORIGINE
4020 O1 = 140:O2 = 95
4025  REM TRACE DE LA SPIRALE
4030  FOR I = 0 TO 220 STEP .01
4040 X = O1 + (I / 2.5) *  COS (I)
4050 Y = O2 + (I / 2.5) *  SIN (I)
4060  GOSUB 3000: REM * POINT *
4070  NEXT I
4080  GET R$: IF R$ = "" THEN 4080
4090  HOME : TEXT
4100  RETURN : REM ----------------------------------
4110  REM
```



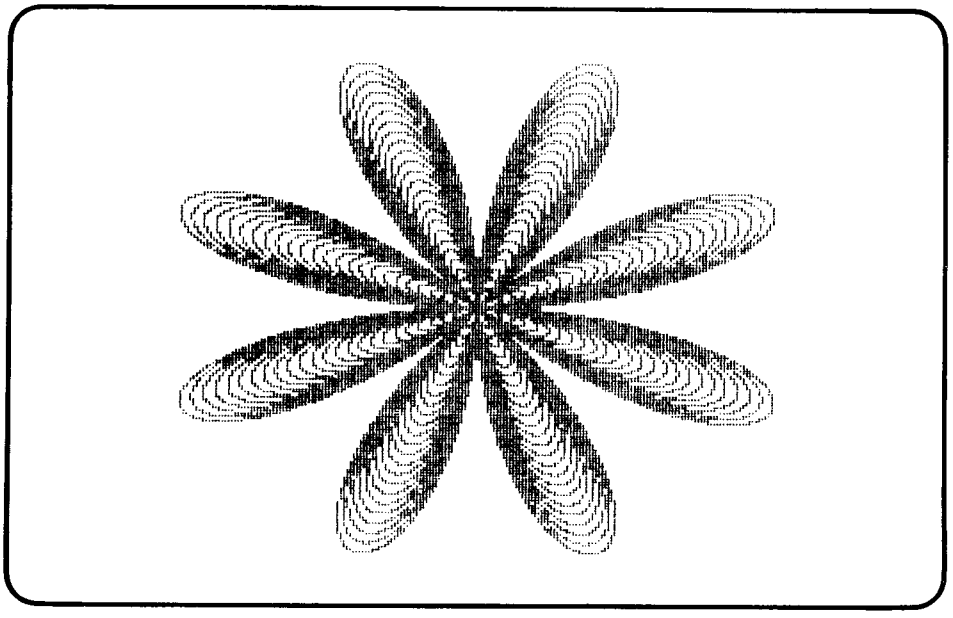
```
$LIST4000-

4000  REM ----------------------------------------
4002  REM FONCTION CIRCULAIRE 4
4004  REM ----------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB = 0:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 96
4035  REM PRECISION DU TRACE
4040 PR = 1 / 850
4045  REM TRACE DE LA FIGURE
4050  FOR T = TB TO TH STEP PR
4060 R = 95 *  SIN (9 * T)
4070 X = O1 + R *  COS (T)
4080 Y = O2 + R *  SIN (T)
4090  GOSUB 3000: REM * POINT *
4100  NEXT T
4110  GET R$: IF R$ = "" THEN 4110
4120  HOME : TEXT
4130  RETURN : REM ----------------------------------
4140  REM
```



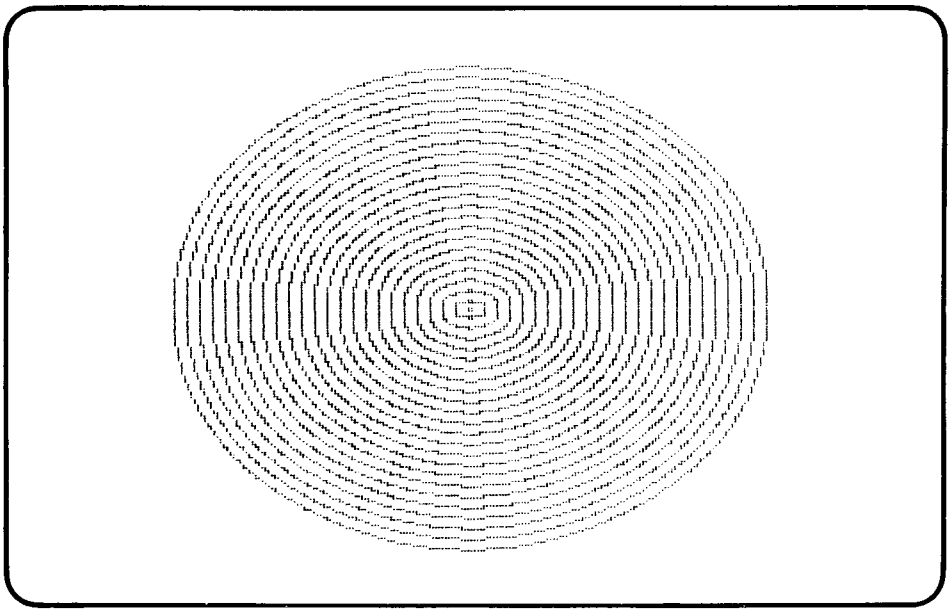
```
§LIST4000-

4000  REM ----------------------------------------
4002  REM FONCTION CIRCULAIRE 5
4004  REM ----------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB =  - 3.14159:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 96
4035  REM PRECISION DU TRACE
4040 PR = 1 / 250
4045  REM TRACE DE LA FIGURE
4050  FOR I = 0 TO 60 STEP 4
4060  FOR T = TB TO TH STEP PR
4070 R = 35 + I *  COS (4 * T)
4080 X = O1 + R *  COS (T)
4090 Y = O2 + R *  SIN (T)
4100  GOSUB 3000: REM * POINT *
4110  NEXT T
4120  NEXT I
4130  GET R$: IF R$ = "" THEN 4130
4140  HOME : TEXT
4150  RETURN : REM ----------------------------------
4160  REM
```



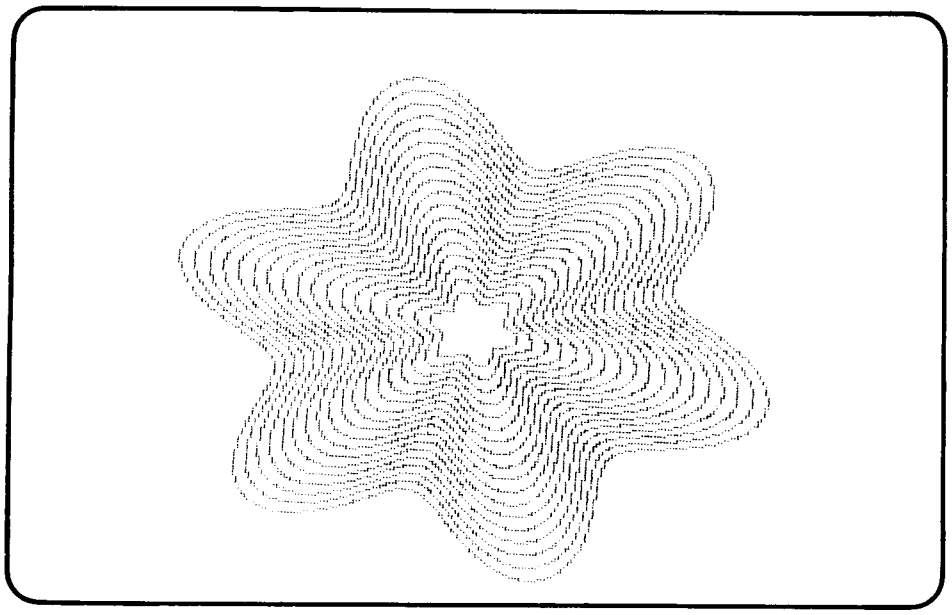
```
$LIST4000-

4000  REM ------------------------------------------------------------
4002  REM FONCTION CIRCULAIRE 6
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB =  - 3.14159:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 96
4035  REM PRECISION DU TRACE
4040 PR = 1 / 400
4045  REM TRACE DE LA FIGURE
4050  FOR I = 8 TO 100 STEP 4
4060  FOR T = TB TO TH STEP PR
4070 R = I *  SIN (4 * T)
4080 X = O1 + R *  COS (T)
4090 Y = O2 + R *  SIN (T)
4100  GOSUB 3000: REM * POINT *
4110  NEXT T
4120  NEXT I
4130  GET R$: IF R$ = "" THEN 4130
4140  HOME : TEXT
4150  RETURN : REM ------------------------------------------------------
4160  REM
```



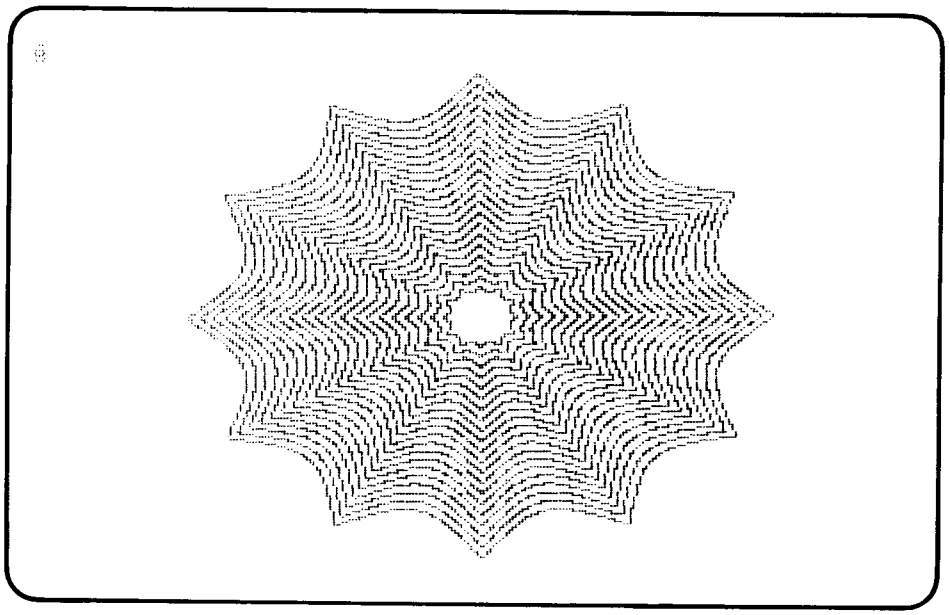
```
§LIST4000-

4000  REM ----------------------------------------------------------------
4002  REM FONCTION CIRCULAIRE 7
4004  REM ----------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB =  - 3.14159:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 96
4035  REM PRECISION DU TRACE
4040 PR = 1 / 100
4045  REM TRACE DE LA FIGURE
4050  FOR I = 0 TO 94 STEP 4
4060  FOR T = TB TO TH STEP PR
4070 R =  SIN (2 *  COS (2 * T)) + I
4080 X = O1 + R *  COS (T)
4090 Y = O2 + R *  SIN (T)
4100  GOSUB 3000: REM * POINT *
4110  NEXT T
4120  NEXT I
4130  GET R$: IF R$ = "" THEN 4130
4140  HOME : TEXT
4150  RETURN : REM ----------------------------------------------------------
4160  REM
```



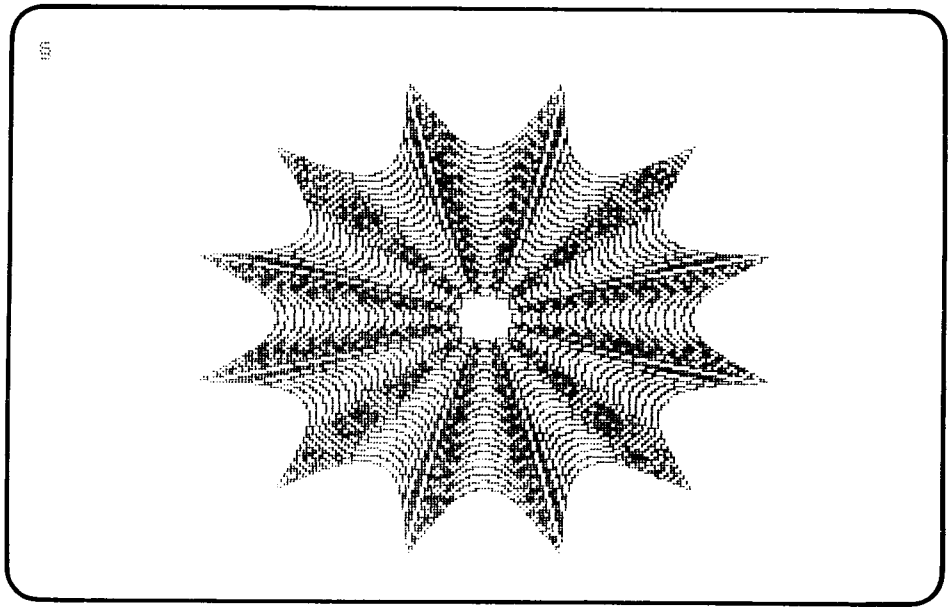
```
$LIST4000-

4000  REM ------------------------------------------------------------
4002  REM FONCTION CIRCULAIRE 8
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB =  - 3.14159:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 96
4035  REM PRECISION DU TRACE
4040 PR = 1 / 125
4045  REM TRACE DE LA FIGURE
4050  FOR I = 12 TO 80 STEP 4
4060  FOR T = TB TO TH STEP PR
4070 R = I * (1 + (.2 *  SIN (6 * T)))
4080 X = O1 + R *  COS (T)
4090 Y = O2 + R *  SIN (T)
4100  GOSUB 3000: REM * POINT *
4110  NEXT T
4120  NEXT I
4130  GET R$: IF R$ = "" THEN 4130
4140  HOME : TEXT
4150  RETURN : REM ------------------------------------------------------
4160  REM
```



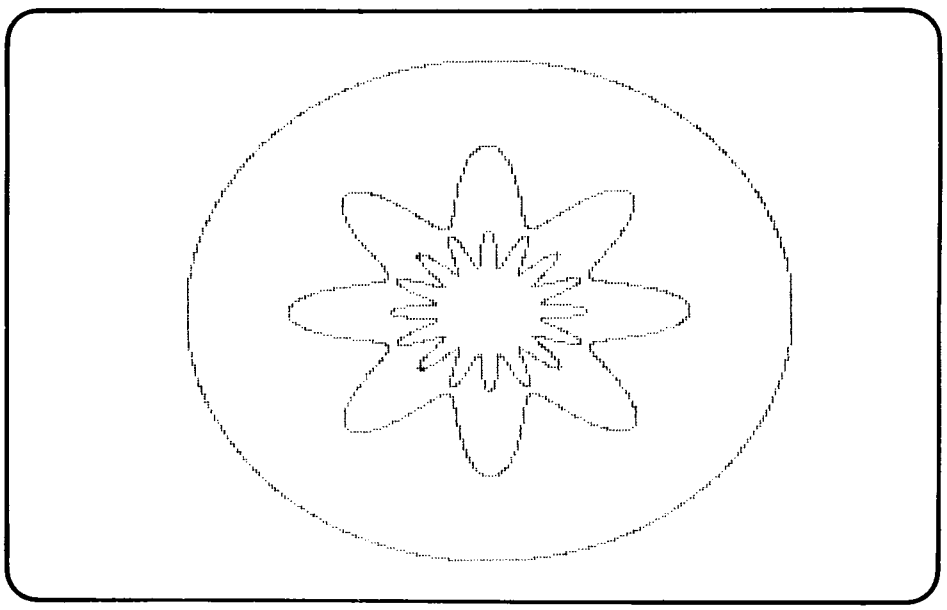
```
§LIST4000-

4000  REM ------------------------------------------------------------
4002  REM FONCTION CIRCULAIRE 9
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB =  - 3.14159:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 96
4035  REM PRECISION DU TRACE
4040 PR = 1 / 150
4045  REM TRACE DE LA FIGURE
4050  FOR I = 12 TO 92 STEP 4
4060  FOR T = TB TO TH STEP PR
4070 R = I / (1 + .2 *  ABS ( SIN (6 * T)))
4080 X = O1 + R *  COS (T)
4090 Y = O2 + R *  SIN (T)
4100  GOSUB 3000: REM * POINT *
4110  NEXT T
4120  NEXT I
4130  GET R$: IF R$ = "" THEN 4130
4140  HOME : TEXT
4150  RETURN : REM ------------------------------------------------------
4160  REM
```



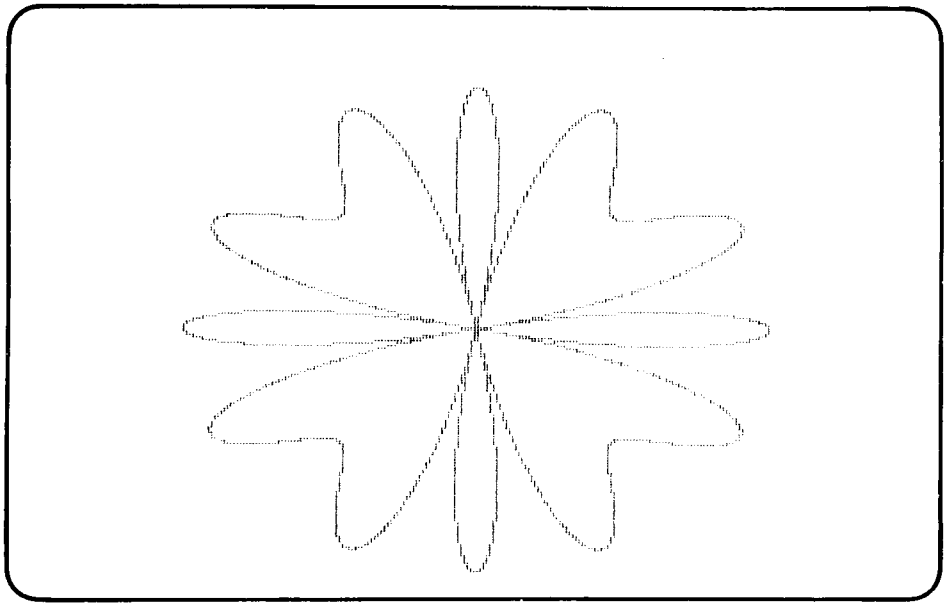
```
$LIST4000-

4000  REM ------------------------------------------------
4002  REM FONCTION CIRCULAIRE·10
4004  REM ------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB =  - 3.14159:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 96
4035  REM PRECISION DU TRACE
4040 PR = 1 / 250
4045  REM TRACE DE LA FIGURE
4050  FOR I = 12 TO 92 STEP 4
4060  FOR T = TB TO TH STEP PR
4070 R = I / (1 + .5 *  ABS ( SIN ( COS (6 * T))))
4080 X = O1 + R *  COS (T)
4090 Y = O2 + R *  SIN (T)
4100  GOSUB 3000: REM * POINT *
4110  NEXT T
4120  NEXT I
4130  GET R$: IF R$ = "" THEN 4130
4140  HOME : TEXT
4150  RETURN : REM ----------------------------------------
4160  REM
```



```
§LIST4000-

4000  REM ------------------------------------------------------------
4002  REM FONCTION CIRCULAIRE 11
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB =  - 3.14159:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 95
4035  REM PRECISION DU TRACE
4040 PR = 1 / 150
4045  REM TRACE DE LA FIGURE
4050  FOR I = 0 TO 8 STEP 4
4060  FOR T = TB TO TH STEP PR
4070 R = (95 - 8 * I) *  COS ( SIN (I * T))
4080 X = O1 + R *  COS (T)
4090 Y = O2 + R *  SIN (T)
4100  GOSUB 3000: REM * POINT *
4110  NEXT T
4120  NEXT I
4130  GET R$: IF R$ = "" THEN 4130
4140  HOME : TEXT
4150  RETURN : REM ------------------------------------------------------
4160  REM
```



```
]LIST4000-

4000  REM ------------------------------------------------------------
4002  REM FONCTION CIRCULAIRE 12
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM ANGLE DE ROTATION
4020 TB =  - 3.14159:TH = 3.14159
4025  REM ORIGINE DU REPERE
4030 O1 = 140:O2 = 95
4035  REM PRECISION DU TRACE
4040 PR = 1 / 700
4045  REM TRACE DE LA FIGURE
4050  FOR T = TB TO TH STEP PR
4060 R = 92 *  COS (4 *  SIN (2 * T))
4070 X = O1 + R *  COS (T)
4080 Y = O2 + R *  SIN (T)
4090  GOSUB 3000: REM * POINT *
4100  NEXT T
4110  GET R$: IF R$ = "" THEN 4110
4120  HOME : TEXT
4130  RETURN : REM ------------------------------------------------------
4140  REM
```



Troisième partie

EXPRESSION GRAPHIQUE PAR L'OUTIL

Transformations graphiques

TRANSLATION D'UN POINT

Notre objectif est de représenter une scène et pour cela, nous allons mettre en oeuvre un certain nombre de transformations qui vont nous faciliter la tâche. La première de ces transformations et la **translation** et, avant de voir comment nous allons mettre une surface en translation, nous allons effectuer la translation d'un point.

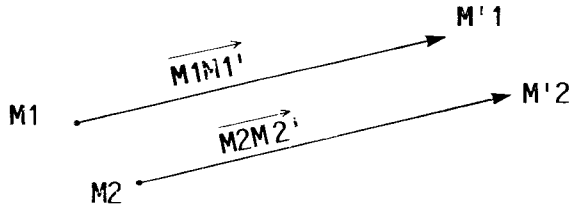
HYPOTHESE

Souhaitant observer la translation d'un point, nous allons considérer l'écran comme muni du même repère que celui utilisé lors du chapitre concernant le point dans les représentations graphiques de base.

Avant de mettre en oeuvre la translation, donnons en une définition :

C'est une transformation géométrique qui à tout point M associe un point M' tels que les vecteurs définis par les couples de points (M,M') sont égaux en sens, en direction et en distance.

Ainsi, si nous considérons la transformation de deux points **M1** et **M2** par la même translation, nous obtenons les points **M1'** et **M2'** comme sur le schéma suivant :



MODELE

En deux dimensions, c'est-à-dire dans le plan, un point est représenté par deux coordonnées **(X,Y)** comme nous l'avons vu. Effectuer la translation, revient alors à additionner une constante à chacune des coordonnées, ce que l'on écrit :

$$X' = X + Tx$$
$$Y' = Y + Ty$$

$$M' = M + Translation$$

ce que l'on écrit sous forme matricielle :

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} + \begin{bmatrix} Tx \\ Ty \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix} + \begin{bmatrix} Tx \\ Ty \end{bmatrix} = \begin{bmatrix} X+Tx \\ Y+Ty \end{bmatrix}$$

ou encore sous une forme plus concise :

$$\begin{bmatrix} X' \\ Y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & Tx \\ 0 & 1 & Ty \\ & & \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} X+Tx \\ Y+Ty \\ 1 \end{bmatrix}$$

Un sous-programme utilisant les coordonnées d'un point pour y ajouter les constantes **Tx** et **Ty** réaliserait une translation et c'est donc cette routine que nous allons écrire.

EXEMPLE D'EXECUTION

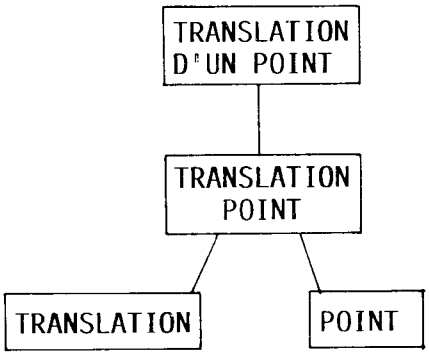
Le programme réalisé et proposé dans ce chapitre permet d'observer la translation d'un point fixe tel que le montre la copie de l'écran de la page suivante.

Dans l'exemple choisi le point fixe se trouve en coordonnées (20,0) et est soumis à une translation (60,80) qui l'amène logiquement en position (20+60,0+80) = (80,80). Le programme décrit la mise en oeuvre d'une translation, et si le processus de cette transformation reste obscur malgré l'exemple d'exécution, il est possible, en modifiant les valeurs des variables XT et YT du sous-programme ***TRANSLATION POINT***, d'étudier d'autres translations.

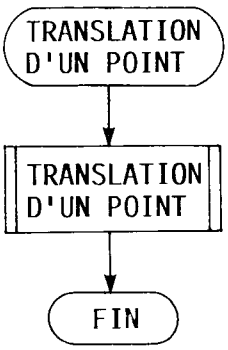
LE PROGRAMME

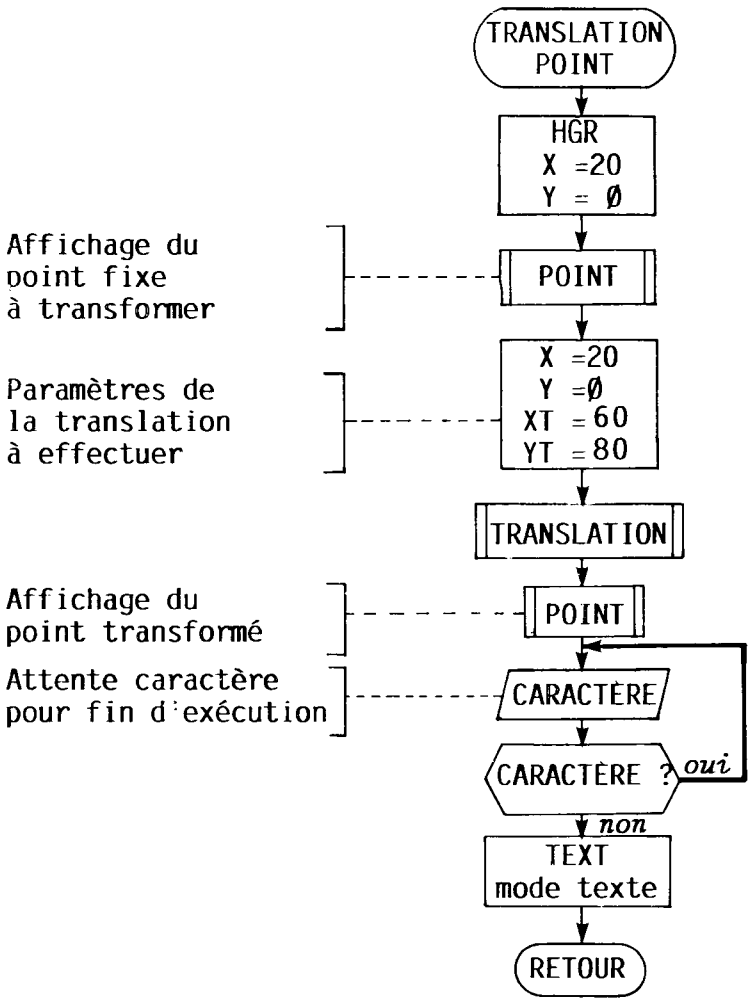
Traitement

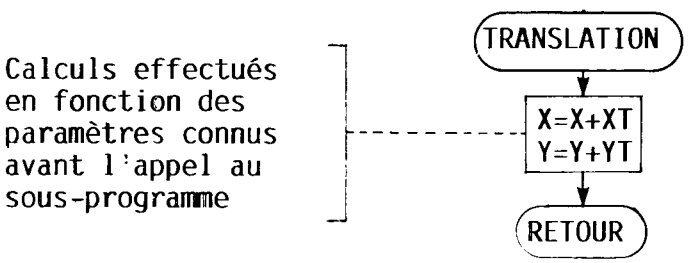
Structure



Organigrammes




TRANSLATION POINT
HGR
X =20
Y = Ø
Affichage du point fixe à transformer
POINT
Paramètres de la translation à effectuer
X =20
Y =Ø
XT = 60
YT = 80
TRANSLATION
Affichage du point transformé
POINT
Attente caractère pour fin d'exécution
CARACTÈRE
CARACTÈRE ?
oui
non
TEXT
mode texte
RETOUR


TRANSLATION
Calculs effectués en fonction des paramètres connus avant l'appel au sous-programme
X=X+XT
Y=Y+YT
RETOUR

Variables

POINT : Affichage d'un point
X : Abscisse du point à afficher
Y : Ordonnée du point à afficher

TRANSLATION : Translation d'un point fixé

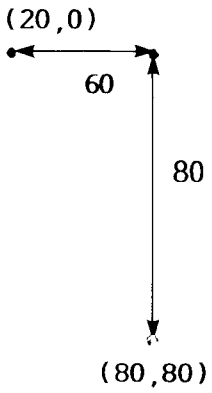
XT : Translation parallèlement à l'axe des abscisses
YT : Translation parallèlement à l'axe des ordonnées
X,Y : En entrée : coordonnées du point à transformer
En sortie : coordonnées du point transformé

TRANSLATION POINT : Affichage d'un point avant et après une translation

X,Y : Coordonnées du point avant et après translation
XT,YT : Valeurs de la translation
R$: Variable pour le caractère de fin d'exécution

```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRANSLATION D'UN POINT
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 5000: REM * TRANSLATION POINT *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
3000  REM ------------------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ---------------------------------------------------
3070  REM
3080  REM
4000  REM ------------------------------------------------------------
4002  REM TRANSLATION
4004  REM ------------------------------------------------------------
4010 X = X + XT
4020 Y = Y + YT
4030  RETURN : REM ---------------------------------------------------
4040  REM
4050  REM
5000  REM ------------------------------------------------------------
5002  REM TRANSLATION POINT
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM POINT FIXE
5020 X = 20:Y = 0
5030  GOSUB 3000: REM * POINT *
5035  REM POINT APRES TRANSLATION
5040 X = 20:Y = 0:XT = 60:YT = 80
5050  GOSUB 4000: REM * TRANSLATION *
5060  GOSUB 3000: REM * POINT *
5070  GET R$: IF R$ = "" THEN 5070
5080  HOME : TEXT
5090  RETURN : REM ---------------------------------------------------
5100  REM
```

Translation d'un point



(20,20) Point avant translation
(80,80) Point après translation

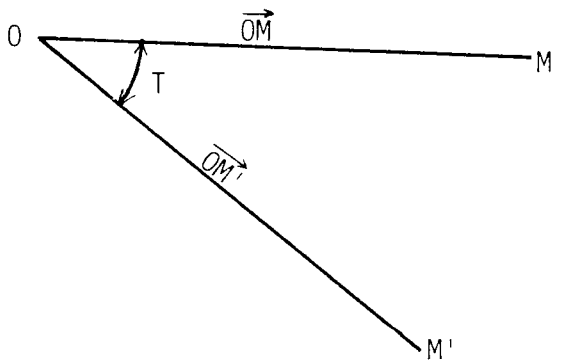
ROTATION D'UN POINT

La seconde transformation élémentaire que nous allons envisager est la rotation d'un point que nous généraliserons par la suite, à à la rotation d'un polygone.

HYPOTHESE

Nous munissons l'écran de notre ordinateur individuel du même repère que celui utilisé pour le chapitre précédent concernant la translation. Puis, nous définissons **la rotation de centre O et d'angle T comme la transformation qui à tout point M associe le point M' tel que la longueur de O à M est identique à celle de O à M' et tel que l'angle entre les vecteurs $\overrightarrow{OM}$ et $\overrightarrow{OM'}$, soit égal à T.**

Une telle rotation peut être représentée comme suit :



MODELE

Notre espace de travail, le plan de l'écran, est muni d'un repère dont nous prenons l'origine pour centre de rotation. En reprenant les notations utilisées dans le chapitre précédent, une rotation d'angle **T** de l'axe **OX** positif à l'axe **OY** positif s'écrit à l'aide des fonctions trigonométriques sinus et cosinus :

$$\begin{cases} X' = X\cos(T) - Y\sin(T) \\ Y' = X\sin(T) + Y\cos(T) \end{cases} \quad (1)$$

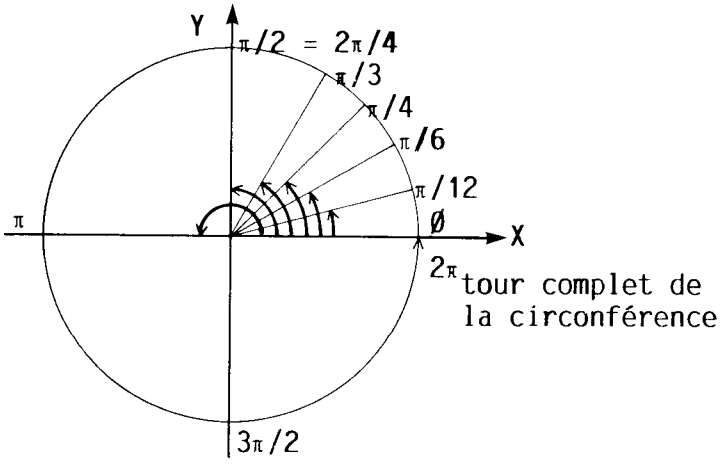
ou encore, sous forme matricielle :

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} \cos(T) & -\sin(T) \\ \sin(T) & \cos(T) \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix}$$

qui donne bien une expression identique à (1) puisque en développant le calcul, on obtient :

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} X\cos(T) - Y\sin(T) \\ X\sin(T) + Y\cos(T) \end{bmatrix}$$

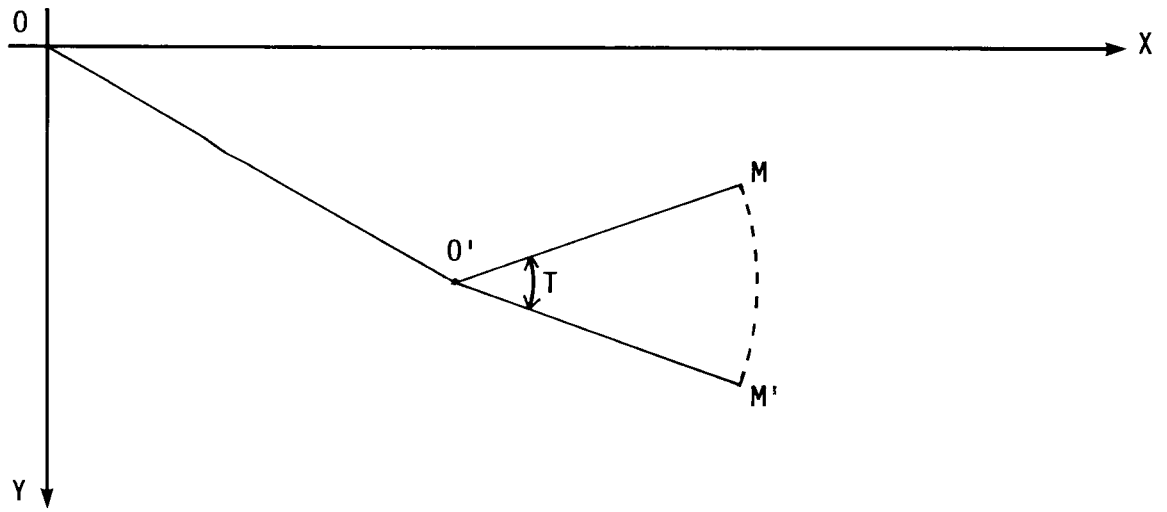
L'angle **T** utilisé par les fonctions trigonométriques doit être précisé en radians car c'est cette unité d'angle qui est utilisée par BASIC. Le schéma ci-dessous rappelle le mode d'évaluation des angles :



Etant donné le calcul à effectuer afin de réaliser la rotation d'un point, nous allons écrire un sous-programme qui, en fonction d'un angle et des coordonnées du point à transformer, calculera le nouveau point.

Jusqu'à présent, nous n'avons envisagé que les rotations dont le centre est l'origine du repère dont est muni l'écran. Or, il peut être intéressant de se donner le choix du centre de rotation. Pour cela, nous allons effectuer un changement de repère, mais auparavant, traçons le schéma du problème afin de mieux cerner sa résolution.

Appelons : **O** L'origine du repère lié à l'écran
O' Le centre de rotation de coordonnées 01 et 02
M Le point à transformer de coordonnées X et Y
M' Le point transformé par la rotation de centre O' et d'angle T ; de coordonnées X' et Y'.



Nous avons, en notation vectorielle : $\overrightarrow{OM'} = \overrightarrow{OO'} + \overrightarrow{O'M'}$

Or, $\overrightarrow{OM'}$ est la rotation d'angle **T** et de centre **O'** du vecteur $\overrightarrow{OM}$ ce qui s'écrit, en notation matricielle :

$$\overrightarrow{O'M'} = \begin{bmatrix} \cos(T) & -\sin(T) \\ \sin(T) & \cos(T) \end{bmatrix} \begin{bmatrix} X - 01 \\ Y - 02 \end{bmatrix}$$

D'où, en remplaçant dans la première équation :

$$\overrightarrow{OM'} = \begin{bmatrix} \cos(T) & -\sin(T) \\ \sin(T) & \cos(T) \end{bmatrix} \begin{bmatrix} X - 01 \\ Y - 02 \end{bmatrix} + \begin{bmatrix} 01 \\ 02 \end{bmatrix}$$

Les équations permettant de calculer les coordonnées **X'** et **Y'** du point **M'** résultat de la rotation d'angle **T** et de centre **O'** du point **M** de coordonnées **X** et **Y** sont les suivantes :

$$X' = (X - 01)\cos(T) - (Y - 02)\sin(T) + 01$$
$$Y' = (X - 01)\sin(T) + (Y - 02)\cos(T) + 02$$

Le second programme que nous allons écrire sera donc quasi-identique à celui que nous avons décrit précédemment cependant, il utilisera en supplément les coordonnées du centre de rotation.

EXEMPLE D'EXECUTION

Nous avons évoqué deux types de rotation et pour chacun d'eux, un programme a été écrit.

Le premier programme effectue la rotation du point fixe de coordonnées (60,0). L'angle de rotation est de $\pi/2$ et le centre de rotation est l'origine du repère lié à l'écran, or puisque la fonction cosinus associe la valeur 0 à l'angle 90° et la fonction sinus, la valeur 1, nécessairement les coordonnées du point après rotation sont **X = 0** et **Y = 60.** On vérifie d'ailleurs que c'est effectivement le résultat obtenu sur la copie d'écran provenant de l'exécution du premier type de rotation.

Le second programme, quant à lui, effectue une rotation dont le centre est fixé par l'utilisateur. Le point fixe dont nous allons effectuer la rotation est situé en (50,50), le centre de rotation est situé en (30,30) et l'angle de la rotation est de $\pi/2$, alors, en plaçant ces valeurs dans les équations permettant le calcul des coordonnées du nouveau point, on obtient :

$$X = (50-30)\ \cos(\pi/2) - (50 - 30)\ \sin(\pi/2) + 30$$
$$Y = (50-30)\ \sin(\pi/2) + (50 - 30)\ \cos(\pi/2) + 30$$

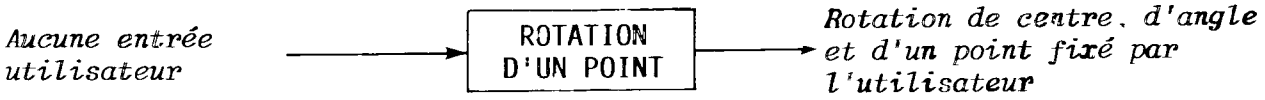
ou encore :

$$X = 20x0 - (20x(+1) + 30 = 10$$
$$Y = 20x(+1) + 20x0 + 30 = 50$$

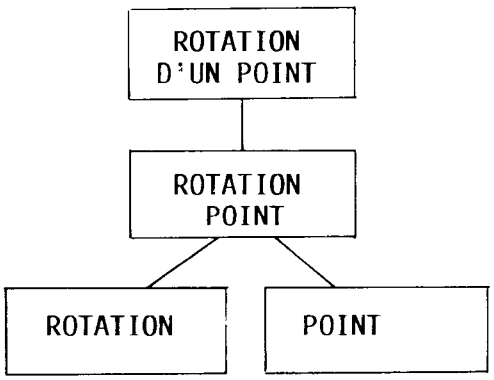
Les coordonnées du point après rotation sont donc (10,50) comme nous pouvons le vérifier sur la copie d'écran du deuxième exemple d'exécution.

LE PROGRAMME

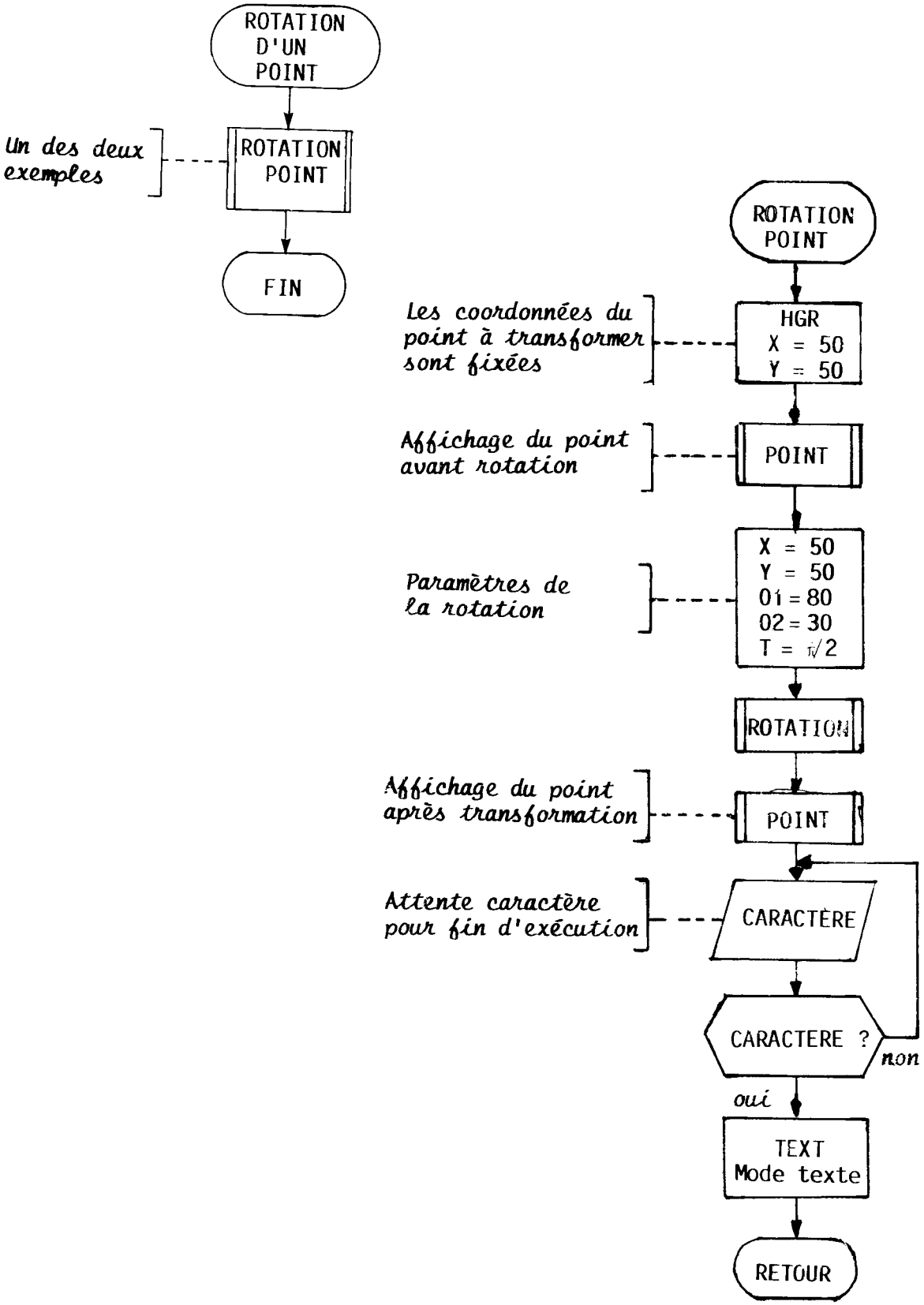
Traitement



Structure



Organigrammes



Variables

POINT	:	Affichage d'un point
X	:	Abscisse du point à afficher
Y	:	Ordonnée du point à afficher
ROTATION	:	Rotation d'un point
T	:	Angle de rotation en radians
01,02	:	Coordonnées du centre de rotation
XP,YP	:	Coordonnées du point après rotation
ROTATION POINT	:	Exemple de rotation d'un point fixe
X,Y	:	Coordonnées du point fixe avant et après rotation
01,02	:	Coordonnées du centre de rotation fixé par l'utilisateur
T	:	Angle de rotation fixé par l'utilisateur
R$	:	Variable pour le caractère de fin d'exécution

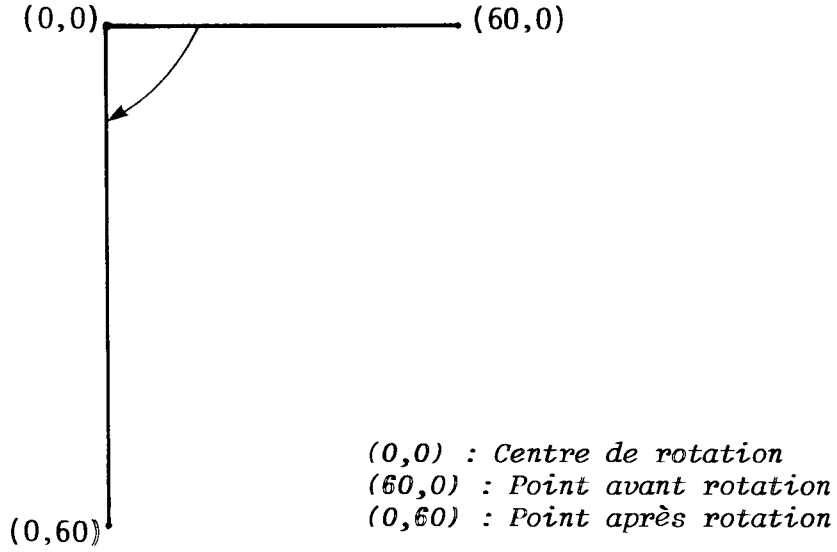
Particularités

Nous n'avons pas encore soulevé le problème du temps d'exécution, pourtant, certains programmes auraient dû être optimisés afin d'obtenir des tracés beaucoup plus rapides.

Dans le programme effectuant la rotation d'un point de centre fixé par l'utilisateur, par exemple, la première amélioration à mettre en oeuvre consisterait à éliminer les calculs redondants tels que (X-01) qui est évalué deux fois pour un point ou encore tels que l'évaluation des valeurs des fonctions trigonométriques.

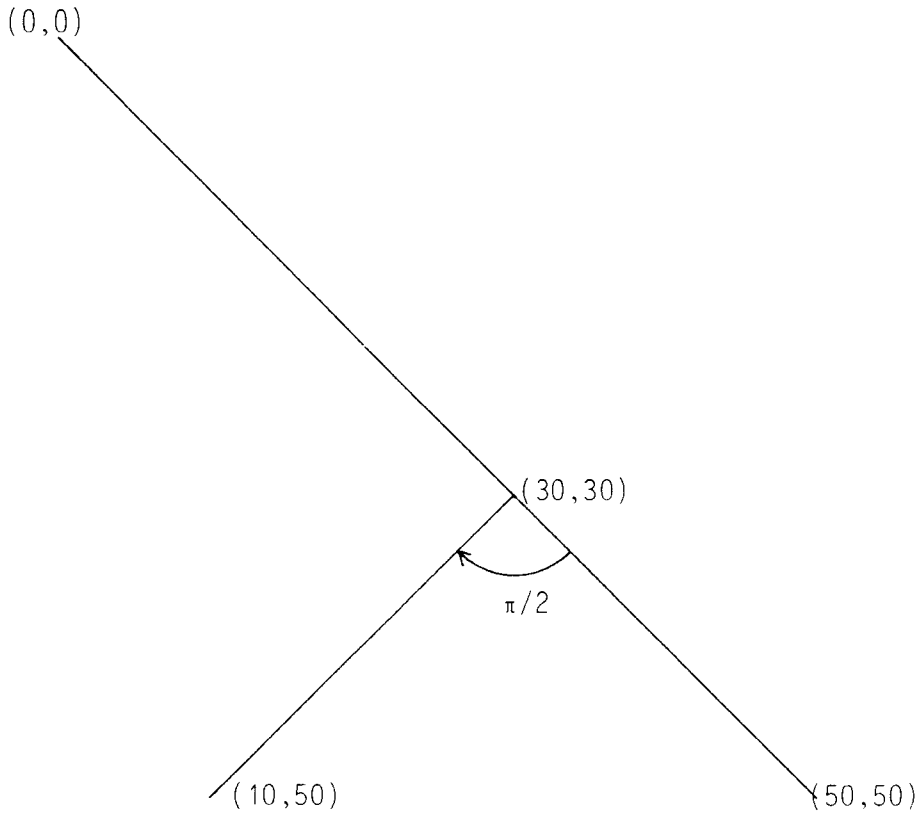
```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM ROTATION D'UN POINT 1
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 5000: REM * ROTATION POINT *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
3000  REM ------------------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ---------------------------------------------------
3070  REM
3080  REM
4000  REM ------------------------------------------------------------
4002  REM ROTATION
4004  REM ------------------------------------------------------------
4010 XP = X *  COS (T) - Y *  SIN (T)
4020 YP = X *  SIN (T) + Y *  COS (T)
4030 X = XP:Y = YP
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM ROTATION POINT
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM POINT FIXE
5020 X = 60:Y = 0
5030  GOSUB 3000: REM * POINT *
5035  REM POINT APRES ROTATION
5040 X = 60:Y = 0:T = 3.14159 / 2
5050  GOSUB 4000: REM * ROTATION *
5060  GOSUB 3000: REM * POINT *
5070  GET R$: IF R$ = "" THEN 5070
5080  HOME : TEXT
5090  RETURN : REM ---------------------------------------------------
5100  REM
```

Exemple de rotation d'un point 1



```
§LIST1000-

1000  REM ------------------------------------------------------------
1002  REM ROTATION D'UN POINT 2
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 5000: REM * ROTATION POINT *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
3000  REM ------------------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ---------------------------------------------------
3070  REM
3080  REM
4000  REM ------------------------------------------------------------
4002  REM ROTATION
4004  REM ------------------------------------------------------------
4010 XP = (X - 01) *  COS (T) - (Y - 02) *  SIN (T) + 01

4020 YP = (X - 01) *  SIN (T) + (Y - 02) *  COS (T) + 02

4030 X = XP:Y = YP
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM ROTATION POINT
5004  REM ------------------------------------------------------------
5005  REM PAGE 1 GRAPHIQUE
5010  HGR : POKE  - 16302,0: HCOLOR= 3
5015  REM POINT FIXE
5020 X = 50:Y = 50
5030  GOSUB 3000: REM * POINT *
5035  REM POINT APRES ROTATION
5040 X = 50:Y = 50:01 = 30:02 = 30:T = 3.14159 / 2
5050  GOSUB 4000: REM * ROTATION *
5060  GOSUB 3000: REM * POINT *
5070  GET R$: IF R$ = "" THEN 5070
5080  HOME : TEXT
5090  RETURN : REM ---------------------------------------------------
5100  REM
```

Exemple de rotation d'un point 2



(30,30) : Centre de rotation
(50,50) : Point avant rotation
(10,50) : Point après rotation

TRANSLATION D'UN POLYGONE

Nous savons comment effectuer la translation d'un point, or il est asez rare que nous n'ayons qu'un point à transformer. En général, nous avons à transformer une figure complexe, c'est-à-dire un ensemble de points. **Nous allons donc généraliser la translation d'un point à celle d'un polygone.**

HYPOTHESE

Dans le chapitre ***TRANSLATION*** d'un point, nous avons considéré l'écran comme muni d'un repère, nous allons considérer la même chose ici. Dans ce référentiel, nous savons réaliser un certain nombre de représentations graphiques de base telles que le point, le vecteur, le polygone. Ces éléments vont nous permettre de mettre en oeuvre, dans ce chapitre, la translation d'une figure complexe : le polygone.

MODELE

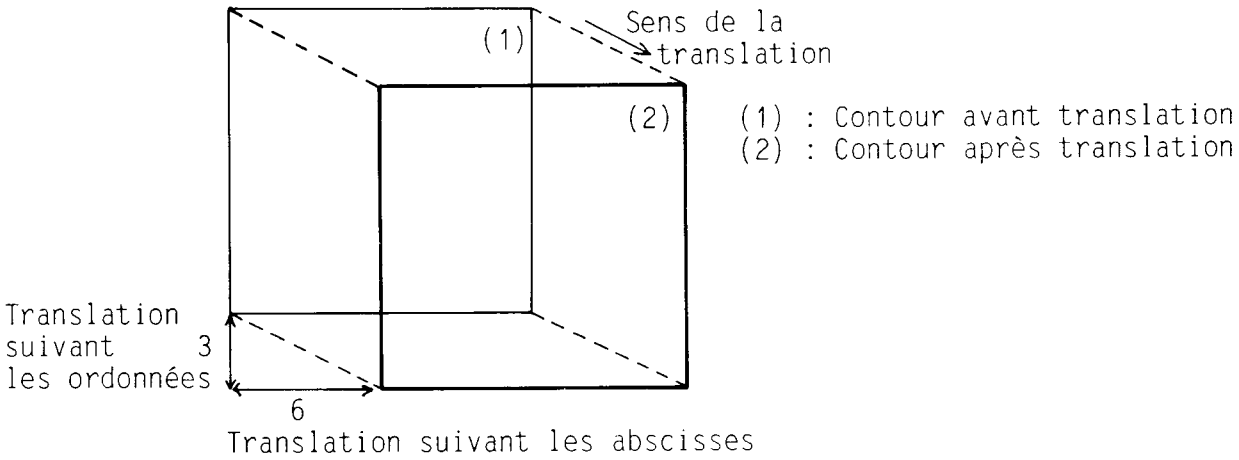
Sachant réaliser la translation d'un point et ayant choisi de définir un polygone comme un ensemble de points, transformer une figure polygonale est très facile. En effet, il suffit d'effectuer la transformation de tous les couples de coordonnées des points pour réaliser la translation du polygone.

En procédant comme indiqué ci-dessus, les modifications à apporter au sous-programme ***TRANSLATION*** , tel que nous l'avions défini, sont nulles. Seule l'organisation du programme change : les sous-programmes ***POLYGONE*** et ***VECTEUR*** sont ajoutés afin de permettre le tracé des figures sur l'écran, avant et après translation.

EXEMPLE D'EXECUTION

Le programme des pages suivantes met en oeuvre la translation d'un polygone. Il effectue en fait, huit transformations successives du même objet. La translation (6,3) consiste en une addition aux valeurs des coordonnées des points du polygone ; cela permet de réaliser le dégradé de représentations que nous pouvons observer sur la copie d'écran de la page

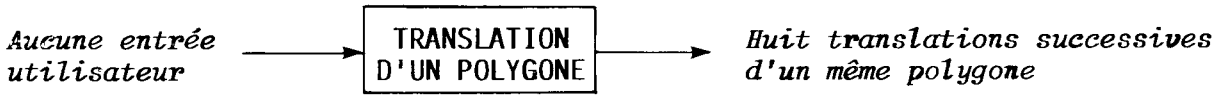
Nous avons pris l'option de transformer successivement plusieurs fois le même polygone. Néanmoins, si nous n'avions effectué qu'une translation, nous aurions abouti à la représentation suivante sur laquelle nous trouvons aussi des informations complémentaires.



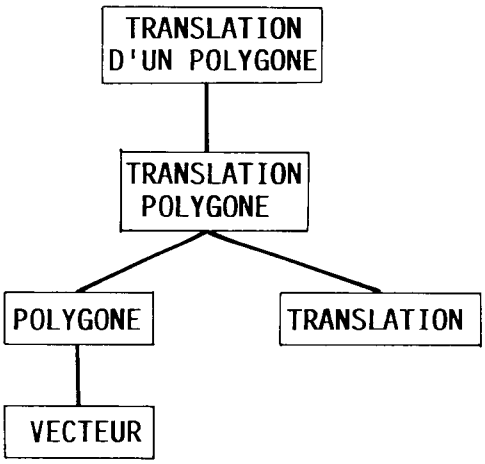
Notons enfin que la translation totale effectuée sur l'exemple d'exécution est égale à la somme des translations. Cette propriété est intéressante car si nous n'avions voulu afficher que deux polygones, celui qui est tracé avant la transformation et celui qui résulte des huits translations, il aurait suffit de ne transformer qu'une fois les points du polygone en multipliant par huit les valeurs de **XT** et **YT**, ce qui revenait à réaliser la translation (48,18)

LE PROGRAMME

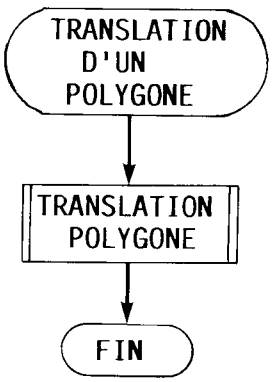
Traitement

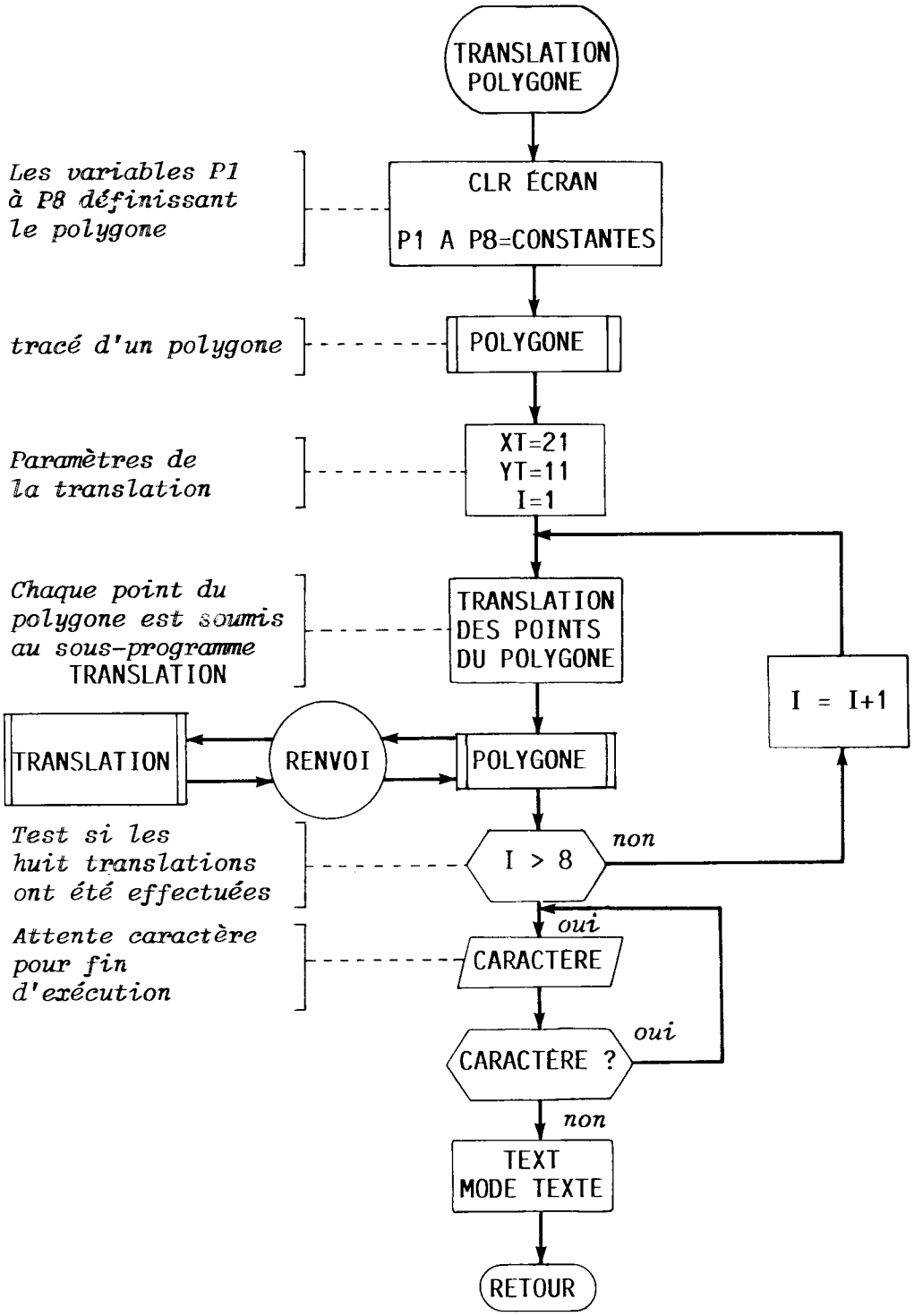


Structure



Organigrammes




TRANSLATION POLYGONE
Les variables P1 à P8 définissant le polygone
CLR ÉCRAN
P1 A P8=CONSTANTES
tracé d'un polygone
POLYGONE
Paramètres de la translation
XT=21
YT=11
I=1
Chaque point du polygone est soumis au sous-programme TRANSLATION
TRANSLATION DES POINTS DU POLYGONE
I = I+1
TRANSLATION
RENVOI
POLYGONE
Test si les huit translations ont été effectuées
I > 8
non
oui
Attente caractère pour fin d'exécution
CARACTERE
CARACTERE ?
oui
non
TEXT
MODE TEXTE
RETOUR

Variables

VECTEUR	:	Tracé d'un vecteur sur l'écran
V1,V2		Coordonnées du premier point à tracer
V3,V4		Coordonnées du second point à tracer
POLYGONE	:	Tracé d'un polygone sur l'écran
P1...P8		Coordonnées des différents points du polygone à tracer
V1...V4		Coordonnées des points des vecteurs à tracer
TRANSLATION	:	Translation d'un point
XT,YT		Valeurs de la translation suivant les deux axes de coordonnées
X,Y		Coordonnées du point avant et après translation
TRANSLATION POLYGONE	:	Tracé d'un polygone soumis à huit translations
P1...P8		Coordonnées des points du polygone
XT,YT		Paramètres de la translation
I		Compteur de translation
X,Y		Coordonnées d'un point à transformer
R$		Variable pour saisie d'un caractère permettant fin d'exécution du programme

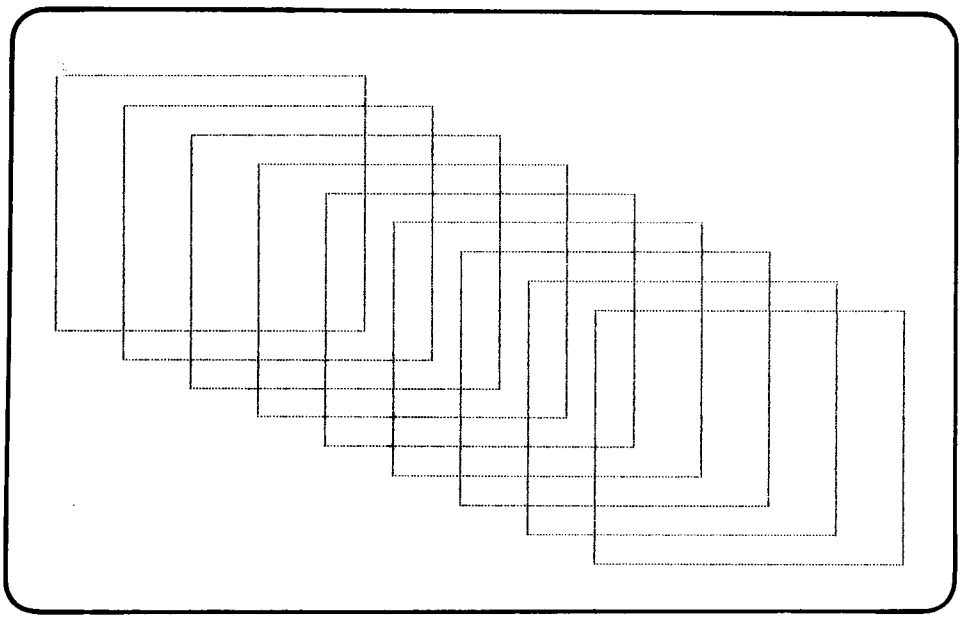
Particularités

Nous avons choisi de conserver le sous-programme translation sous la forme qu'il avait dans le chapitre ***TRANSLATION*** d'un point. Pourtant, dans le cas particulier de ce programme, on aurait pu directement ajouter **XT, YT** aux variables **P1** à **P8.** Cette solution quoique plus performante au niveau du temps d'exécution ne pouvait être envisagée dans un cas général.

. Notons que la translation du polygone telle qu'elle est définie dans le programme, peut faire l'objet d'un sous-programme chargé de la translation des points de coordonnées P1 à P8. Pour les huit translations successives, effectuées dans l'exemple d'exécution, il suffirait alors de huit appels successifs à ce sous-programme. Cette façon de faire apporte une clarté plus grande lorsqu'il faut transformer plusieurs polygones.

```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRANSLATION D'UN POLYGONE
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 7000: REM * TRANSLATION POLYGONE *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM POLYGONE
5004  REM ------------------------------------------------------------
5005  REM (P1,P2)...(P7,P8) : COUPLES DE COORDONNEES
5006  REM DES POINTS DU POLYGONE A TRACER
5010 V1 = P1:V2 = P2:V3 = P3:V4 = P4
5020  GOSUB 4000: REM *VECTEUR*
5030 V1 = P3:V2 = P4:V3 = P5:V4 = P6
5040  GOSUB 4000: REM * VECTEUR *
5050 V1 = P5:V2 = P6:V3 = P7:V4 = P8
5060  GOSUB 4000: REM * VECTEUR *
5070 V1 = P7:V2 = P8:V3 = P1:V4 = P2
5080  GOSUB 4000: REM * VECTEUR *
5090  RETURN : REM ---------------------------------------------------
5100  REM
5110  REM
6000  REM ------------------------------------------------------------
6002  REM TRANSLATION
6004  REM ------------------------------------------------------------
6010 X = X + XT
6020 Y = Y + YT
6030  RETURN : REM ---------------------------------------------------
6040  REM
6050  REM
```

```
7000  REM ----------------------------------------
7002  REM TRANSLATION POLYGONE
7004  REM ----------------------------------------
7005  REM PAGE 1 GRAPHIQUE
7010  HGR : POKE  - 16302,0: HCOLOR= 3
7015  REM POLYGONE DE REFERENCE
7020 P1 = 0:P2 = 0:P3 = 96:P4 = 0
7030 P5 = 96:P6 = 96:P7 = 0:P8 = 96
7040  GOSUB 5000: REM  * POLYGONE *
7045  REM PARAMETRES DE LA TRANSLATION
7050 XT = 21:YT = 11
7055  REM TRANSLATIONS DU POLYGONE
7060  FOR I = 1 TO 8
7070 X = P1:Y = P2
7080  GOSUB 6000: REM * TRANSLATION *
7090 P1 = X:P2 = Y
7100 X = P3:Y = P4
7110  GOSUB 6000: REM * TRANSLATION *
7120 P3 = X:P4 = Y
7130 X = P5:Y = P6
7140  GOSUB 6000: REM * TRANSLATION *
7150 P5 = X:P6 = Y
7160 X = P7:Y = P8
7170  GOSUB 6000: REM * TRANSLATION *
7180 P7 = X:P8 = Y
7190  GOSUB 5000: REM * POLYGONE *
7200  NEXT I
7210  GET R$: IF R$ = "" THEN 7210
7220  HOME : TEXT
7230  RETURN : REM ----------------------------------
7240  REM
```



ROTATION D'UN POLYGONE

Sachant comment réaliser la rotation d'un point, nous allons généraliser le processus à un polygone. Cette nouvelle transformation va alors offrir une grande variété d'effets graphiques que nous allons envisager.

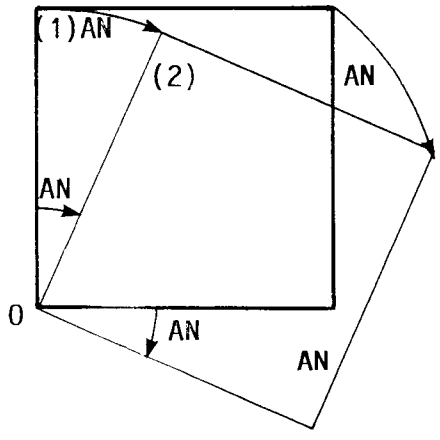
HYPOTHESE

Comme dans le chapitre ***TRANSLATION D'UN POLYGONE***, nous allons utiliser les sous-programmes ***VECTEUR*** et ***POLYGONE*** que nous avons déjà étudiés en détail.

En ce qui concerne la rotation proprement dite, c'est la routine ***ROTATION "de centre fixé par l'utilisateur"*** qui va nous servir. Celle-ci va nous permettre de définir la rotation d'un polygone selon les mêmes critères que ceux que nous avons choisis pour la définition de la translation d'un polygone. Nous resterons ainsi homogènes dans l'approche et la résolution du problème.

MODELE

Nous n'allons pas effectuer la rotation globale d'un polygone mais plutôt exploiter le sous-programme ***ROTATION*** que nous avons déjà mis en oeuvre. Ce sous-programme est adapté à la situation présente, car le polygone est défini comme un ensemble de points et, pour le transformer, il suffit de transformer chacun de ses points comme le prouve le schéma ci-après :



(1) : contour avant rotation
(2) : contour après rotation

AN = angle de rotation
0 = centre de rotation

Pour réaliser la rotation d'un polygone, il suffit donc de calculer la rotation de chacun de ses points et le programme de ce chapitre va mettre en oeuvre cette propriété.

EXEMPLE D'EXECUTION

Un point, mis en rotation, tourne autour d'un centre dont l'importance est toute particulière. Nous avons d'ailleurs été amenés à laisser le choix de la position du centre de rotation à l'utilisateur. Dans le cas de la rotation d'un polygone ou de toute autre représentation graphique complexe à contour fermé, ce choix doit se faire parmi les quatre possibilités que nous allons envisager maintenant.

. Centre à l'origine du repère de l'écran

Le choix du centre de rotation à l'origine du repère lié à l'écran a été fait afin de montrer quel aurait été l'effet produit lors de la rotation d'un polygone si nous avions conservé le premier sous-programme ***ROTATION*** établi dans le chapitre ***ROTATION D'UN POINT***.

Quatre rotations successives sont réalisées comme nous pouvons l'observer sur la première copie d'écran de ce chapitre. Le polygone de départ situé le plus haut est soumis à une rotation d'angle $\pi/8$. Le polygone résultant se trouve alors déformé pour des raisons de résolution d'écran et sert de base pour une nouvelle rotation. Ce processus est répété quatre fois, ce qui permet de bien décomposer la rotation d'angle $\pi/2$, du centre origine du repère lié à l'écran, finalement réalisée.

. Centre extérieur au polygone

Le mode de rotation que nous allons étudier maintenant nécessite les coordonnées d'un centre fixé par l'utilisateur, hors du polygone. Dans l'exemple d'exécution que nous avons réalisé et dont le résultat est donné par la deuxième copie d'écran, le centre de rotation a les coordonnées (40,24). Le polygone défini dans les premières lignes du deuxième sous-programme ***ROTATION POLYGONE***, étant de petite taille, il est possible d'effectuer quinze transformations successives d'angle $\pi/2$. La figure effectue ainsi un tour complet autour du centre qui lui est extérieur.

. Centre sur le contour du polygone

Cette fois, le centre de rotation fixé par l'utilisateur est sur le contour du polygone. Dans le cas précis du troisième exemple d'exécution, il est même sur l'un des coins du polygone. Ce choix permet de réaliser, un effet graphique plus intéressant lorsque les quinze rotations d'angle $\pi/2$ ont été réalisées. La troisième copie d'écran de ce chapitre constitue le compte-rendu de la rotation dont il est question ici, et permet de constater une nouvelle fois l'importance du choix du centre de rotation.

Centre intérieur au contour du polygone

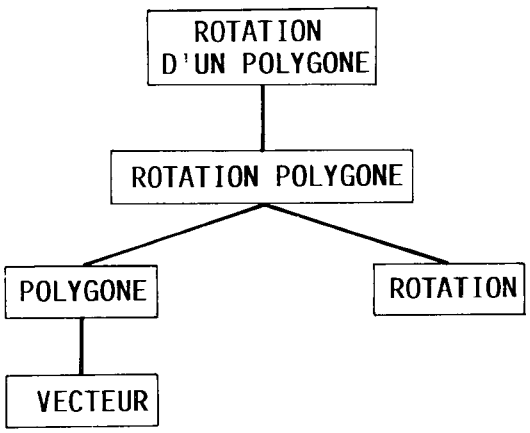
La dernière possibilité que nous allons voir est le choix du centre de rotation à l'intérieur du polygone. La représentation graphique alors obtenue est une fois de plus différente comme le montre la dernière copie d'écran des pages suivantes. L'angle de transformation de $\Pi/8$ permet au polygone de faire un tour complet après quinze rotations successives.

Nous venons de le constater, la rotation offre de nombreuses possibilités graphiques. Elle se généralise facilement à n'importe quel type de représentation et il ne reste qu'à laisser libre cours à son imagination.

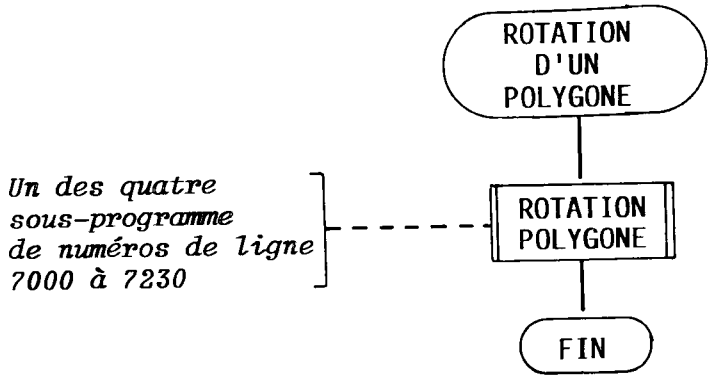
LE PROGRAMME

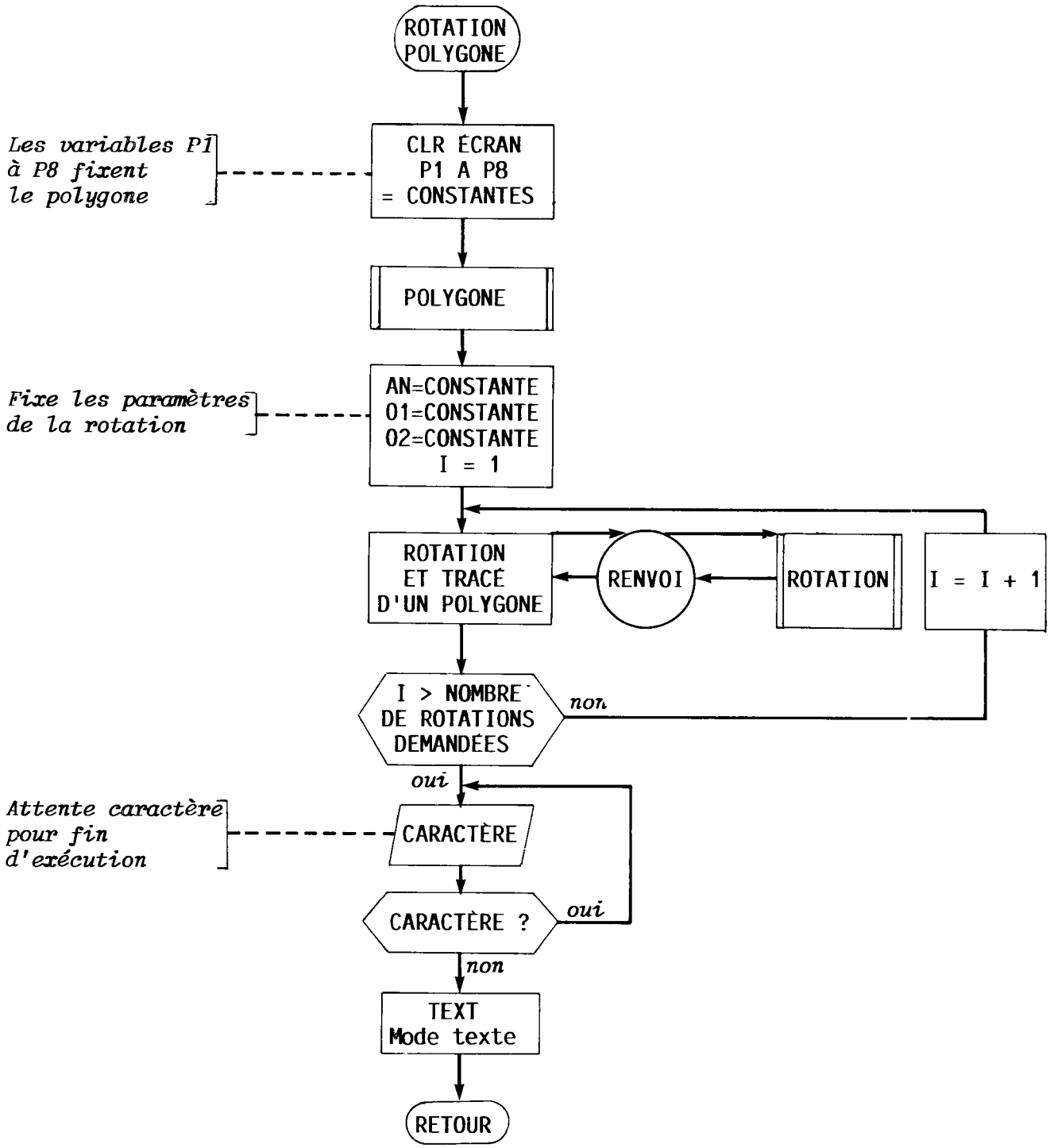
Traitement

Structure



Organigrammes




ROTATION POLYGONE
Les variables P1 à P8 fixent le polygone
CLR ÉCRAN P1 A P8 = CONSTANTES
POLYGONE
Fixe les paramètres de la rotation
AN=CONSTANTE
01=CONSTANTE
02=CONSTANTE
I = 1
ROTATION ET TRACÉ D'UN POLYGONE
RENVOI
ROTATION
I = I + 1
I > NOMBRE DE ROTATIONS DEMANDÉES
non
oui
Attente caractère pour fin d'exécution
CARACTÈRE
CARACTÈRE ?
oui
non
TEXT
Mode texte
RETOUR

Variables

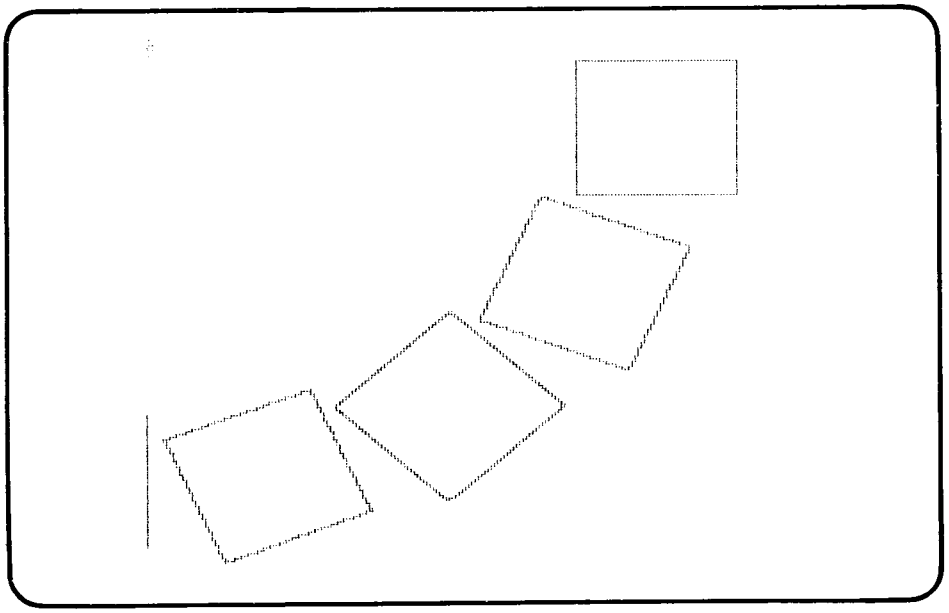
VECTEUR	:	Tracé d'un vecteur sur l'écran
V1,V2	:	Coordonnées du premier point du vecteur à tracer
V3,V4	:	Coordonnées du second point du vecteur à tracer
POLYGONE	:	Tracé d'un polygone sur l'écran
P1...P8	:	Coordonnées des différents points du polygone à tracer
V1...V4	:	Coordonnées des points des vecteurs à tracer
ROTATION	:	Rotation d'un point
01,02	:	Coordonnées du centre de rotation
AN	:	Angle de rotation
XP,YP	:	Coordonnées du point après rotation
X,Y	:	Coordonnées du point avant et après rotation
ROTATION POLYGONE	:	Tracé d'un polygone soumis à plusieurs rotations
P1...P8	:	Coordonnées des points du polygone
AN	:	Angle de la rotation
01,02	:	Coordonnées du centre de rotation
I	:	Compteur de rotation
X,Y	:	Coordonnées du point à transformer
R$	:	Variable pour saisie d'un caractère de la fin d'exécution du programme.

Particularité

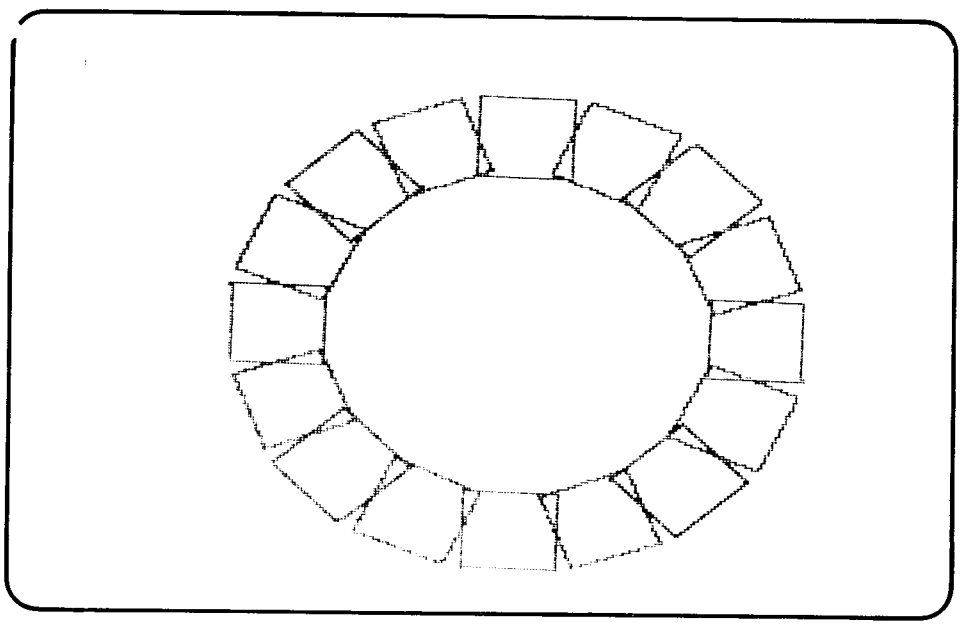
Dans l'ensemble des sous-programmes ***ROTATION POLYGONE*** seules les valeurs des variables P1 à P8 et des paramètres de la rotation changent ; aussi, il suffit de taper les lignes une fois, puis, de modifier les valeurs pour obtenir tous les exemples d'exécution.

```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM ROTATION D'UN POLYGONE 1
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 7000: REM * ROTATION POLYGONE 1 *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM POLYGONE
5004  REM ------------------------------------------------------------
5005  REM (P1,P2)...(P7,P8) : COUPLES DE COORDONNEES
5006  REM DES POINTS DU POLYGONE A TRACER
5010 V1 = P1:V2 = P2:V3 = P3:V4 = P4
5020  GOSUB 4000: REM *VECTEUR*
5030 V1 = P3:V2 = P4:V3 = P5:V4 = P6
5040  GOSUB 4000: REM * VECTEUR *
5050 V1 = P5:V2 = P6:V3 = P7:V4 = P8
5060  GOSUB 4000: REM * VECTEUR *
5070 V1 = P7:V2 = P8:V3 = P1:V4 = P2
5080  GOSUB 4000: REM * VECTEUR *
5090  RETURN : REM ---------------------------------------------------
5100  REM
5110  REM
6000  REM ------------------------------------------------------------
6002  REM ROTATION
6004  REM ------------------------------------------------------------
6010 XP = (X - O1) *  COS (AN) - (Y - O2) *  SIN (AN) +
    O1
6020 YP = (X - O1) *  SIN (AN) + (Y - O2) *  COS (AN) +
    O2
6030 X = XP:Y = YP
6040  RETURN : REM ---------------------------------------------------
6050  REM
6060  REM
```

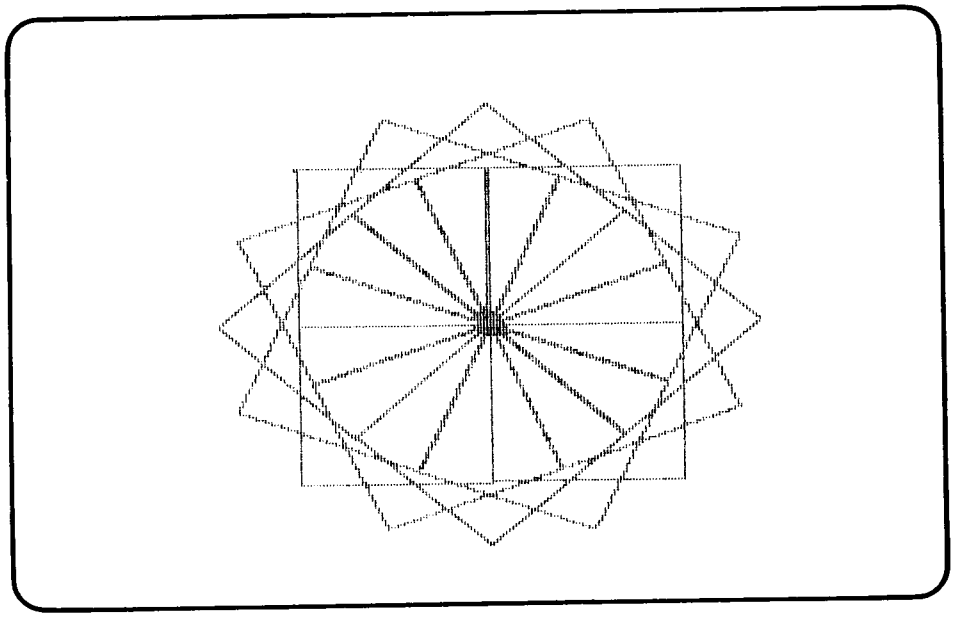
```
7000  REM ------------------------------------------------------------
7002  REM ROTATION POLYGONE 1
7004  REM ------------------------------------------------------------
7005  REM PAGE 1 GRAPHIQUE
7010  HGR : POKE  - 16302,0: HCOLOR= 3
7015  REM POLYGONE DE REFERENCE
7020 P1 = 134:P2 = 0:P3 = 184:P4 = 0
7030 P5 = 184:P6 = 50:P7 = 134:P8 = 50
7040  GOSUB 5000: REM  * POLYGONE *
7045  REM PARAMETRES DE LA ROTATION
7050 AN = 3.14159 / 8:01 = 0:02 = 0
7055  REM ROTATIONS DU POLYGONE
7060  FOR I = 1 TO 4
7070 X = P1:Y = P2
7080  GOSUB 6000: REM * ROTATION *
7090 P1 = X:P2 = Y
7100 X = P3:Y = P4
7110  GOSUB 6000: REM * ROTATION *
7120 P3 = X:P4 = Y
7130 X = P5:Y = P6
7140  GOSUB 6000: REM * ROTATION *
7150 P5 = X:P6 = Y
7160 X = P7:Y = P8
7170  GOSUB 6000: REM * ROTATION *
7180 P7 = X:P8 = Y
7190  GOSUB 5000: REM * POLYGONE *
7200  NEXT I
7210  GET R$: IF R$ = "" THEN 7210
7220  HOME : TEXT
7230  RETURN : REM ------------------------------------------------------
7240  REM
```



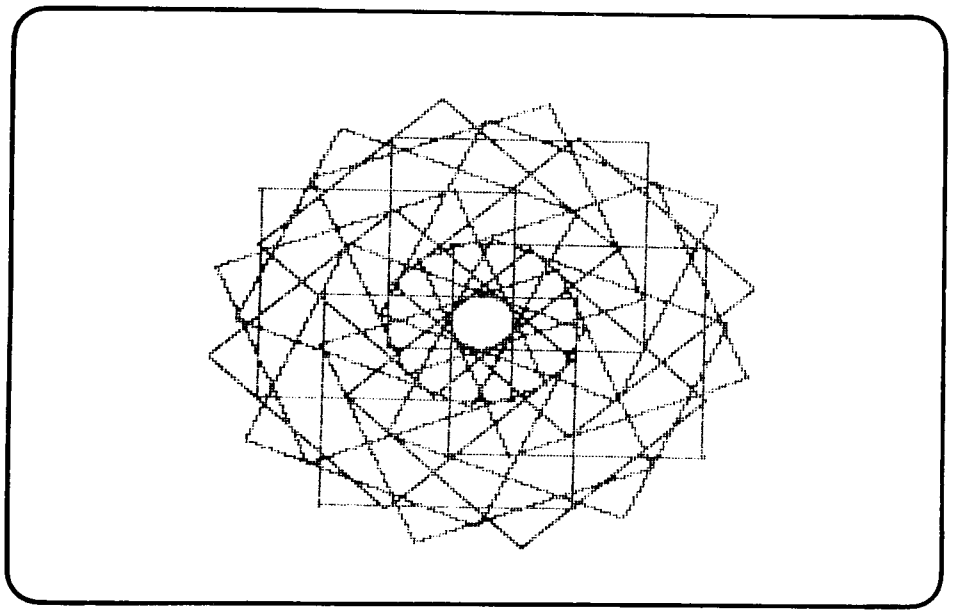
```
]LIST7000-

7000  REM ----------------------------------------------------------
7002  REM ROTATION POLYGONE 2
7004  REM ----------------------------------------------------------
7005  REM PAGE 1 GRAPHIQUE
7010  HGR : POKE  - 16302,0: HCOLOR= 3
7015  REM POLYGONE DE REFERENCE
7020 P1 = 125:P2 = 0:P3 = 155:P4 = 0
7030 P5 = 155:P6 = 30:P7 = 125:P8 = 30
7040  GOSUB 5000: REM  * POLYGONE *
7045  REM PARAMETRES DE LA ROTATION
7050 AN = 3.14159 / 8:01 = 140:02 = 90
7055  REM ROTATIONS DU POLYGONE
7060  FOR I = 1 TO 15
7070 X = P1:Y = P2
7080  GOSUB 6000: REM * ROTATION *
7090 P1 = X:P2 = Y
7100 X = P3:Y = P4
7110  GOSUB 6000: REM * ROTATION *
7120 P3 = X:P4 = Y
7130 X = P5:Y = P6
7140  GOSUB 6000: REM * ROTATION *
7150 P5 = X:P6 = Y
7160 X = P7:Y = P8
7170  GOSUB 6000: REM * ROTATION *
7180 P7 = X:P8 = Y
7190  GOSUB 5000: REM * POLYGONE *
7200  NEXT I
7210  GET R$: IF R$ = "" THEN 7210
7220  HOME : TEXT
7230  RETURN : REM ----------------------------------------------------
7240  REM
```



```
7000  REM ----------------------------------------
7002  REM ROTATION POLYGONE 3
7004  REM ----------------------------------------
7005  REM PAGE 1 GRAPHIQUE
7010  HGR : POKE  - 16302,0: HCOLOR= 3
7015  REM POLYGONE DE REFERENCE
7020 P1 = 140:P2 = 36:P3 = 200:P4 = 36
7030 P5 = 200:P6 = 96:P7 = 140:P8 = 96
7040  GOSUB 5000: REM  * POLYGONE *
7045  REM PARAMETRES DE LA ROTATION
7050 AN = 3.14159 / 8:O1 = 140:O2 = 96
7055  REM ROTATIONS DU POLYGONE
7060  FOR I = 1 TO 15
7070 X = P1:Y = P2
7080  GOSUB 6000: REM * ROTATION *
7090 P1 = X:P2 = Y
7100 X = P3:Y = P4
7110  GOSUB 6000: REM * ROTATION *
7120 P3 = X:P4 = Y
7130 X = P5:Y = P6
7140  GOSUB 6000: REM * ROTATION *
7150 P5 = X:P6 = Y
7160 X = P7:Y = P8
7170  GOSUB 6000: REM * ROTATION *
7180 P7 = X:P8 = Y
7190  GOSUB 5000: REM * POLYGONE *
7200  NEXT I
7210  GET R$: IF R$ = "" THEN 7210
7220  HOME : TEXT
7230  RETURN : REM ----------------------------------
7240  REM
```



```
]LIST7000-

7000  REM ----------------------------------------
7002  REM ROTATION POLYGONE 4
7004  REM ----------------------------------------
7005  REM PAGE 1 GRAPHIQUE
7010  HGR : POKE  - 16302,0: HCOLOR= 3
7015  REM POLYGONE DE REFERENCE
7020 P1 = 110:P2 = 26:P3 = 190:P4 = 26
7030 P5 = 190:P6 = 106:P7 = 110:P8 = 106
7040  GOSUB 5000: REM  * POLYGONE *
7045  REM PARAMETRES DE LA ROTATION
7050 AN = 3.14159 / 8:01 = 140:02 = 96
7055  REM ROTATIONS DU POLYGONE
7060  FOR I = 1 TO 15
7070 X = P1:Y = P2
7080  GOSUB 6000: REM * ROTATION *
7090 P1 = X:P2 = Y
7100 X = P3:Y = P4
7110  GOSUB 6000: REM * ROTATION *
7120 P3 = X:P4 = Y
7130 X = P5:Y = P6
7140  GOSUB 6000: REM * ROTATION *
7150 P5 = X:P6 = Y
7160 X = P7:Y = P8
7170  GOSUB 6000: REM * ROTATION *
7180 P7 = X:P8 = Y
7190  GOSUB 5000: REM * POLYGONE *
7200  NEXT I
7210  GET R$: IF R$ = "" THEN 7210
7220  HOME : TEXT
7230  RETURN : REM ----------------------------------
7240  REM
```



Scènes en deux dimensions

DÉFINITION DE SURFACES

Notre objectif est la représentation d'une scène et, pour cela, nous allons définir des objets. Ces derniers, cependant, se résumeront à des surfaces, car nous travaillons dans le plan défini par l'écran et donc, en deux dimensions. Jusqu'à présent, nous avons mis en oeuvre des programmes permettant le tracé d'objets graphiques non manipulables. Cette fois, **nous allons étudier une méthode pratique de définition de surfaces sous forme polygonale qui nous permettra, par la suite, d'effectuer diverses transformations et donc, de manipuler des objets pour réaliser des scènes variées.**

HYPOTHESE

Nous allons définir les objets, éléments constitutifs d'une scène, sous forme polygonale. En conséquence, nous allons utiliser des vecteurs ou encore, le sous-programme ***VECTEUR*** tel qu'il a été défini dans le chapitre du même nom.

En ce qui concerne la surface de l'écran, nous allons la considérer comme la partie visible de la scène. La fenêtre ainsi définie ne permettra le tracé que des objets intérieurs à l'écran. Pour assurer cette fonction, aucune opération particulière de découpage des segments (clipping) sortant de l'écran n'est nécessaire car le sous-programme ***VECTEUR***, déjà pourvu de tests, ne permet pas le tracé de points ne satisfaisant pas aux conditions limites.

Cette méthode apporte une simplification à la constitution de scènes. Cependant, elle présente un désavantage notable. Lorsqu'un vecteur est entièrement à l'extérieur de la fenêtre de

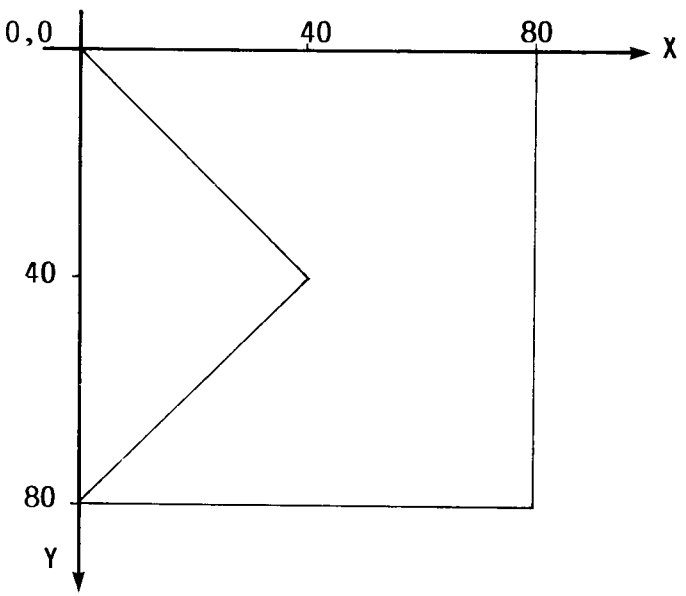
l'écran, il n'est pas tracé et le résultat voulu est obtenu. En revanche, lorsqu'un vecteur possède une partie à l'intérieur et une autre à l'extérieur de la fenêtre, il n'est pas tracé et là, un problème se pose. Pour y remédier, il suffit de remplacer la coordonnée qui franchit la limite qui lui est imposée, par la limite elle-même.

Cette opération n'est pas mise en oeuvre dans les programmes de ce livre afin de rester homogène avec les sous-programmes déjà réalisés mais, gageons que le lecteur n'aura aucun mal à écrire une routine réalisant cette fonction.

MODELE

Pour définir une surface, nous allons considérer uniquement son contour. L'objet, c'est-à-dire la surface sera alors la partie du plan (de l'écran), enfermée par cette frontière. On déduit de cette définition qu'il suffit de connaître les contours c'est-à-dire les segments qui le définissent, pour connaître la surface.

Définissons une surface sur le plan suivant :



Pour définir cette surface, il faut donc connaître l'ensemble des vecteurs qui délimitent son contour. Alors, nous choisissons un sens de parcours du contour et considérons qu'un vecteur est d'abord un couple de points. Cette considération en précède une autre qui consiste à remarquer que chaque point du contour est commun à deux vecteurs. Alors, en complétant la liste des points, ordonnée suivant un sens de parcours choisi, d'un indicateur prenant la valeur un lorsque deux points sont liés et la valeur zéro lorsqu'ils ne le sont pas, on est capable de définir, pour la surface ci-dessus, la liste suivante :

	POINTS	
Indicateur	X	Y
0	0	0
1	40	40
1	0	80
1	80	80
1	80	0
1	0	0

qui définit la surface complètement.

Sur le plan pratique, c'est-à-dire au niveau du programme, lors de la définition de la surface, nous ajoutons en début de liste le nombre total de points à lire afin de pouvoir définir plusieurs surfaces. Pour tracer la surface, nous lisons le premier point dont l'indicateur est obligatoirement à zéro et, nous le considérons comme point courant. Nous lisons ensuite le second point et, suivant son indicateur nous traçons ou ne traçons pas un segment. Puis, nous le considérons comme nouveau point courant et recommençons le processus jusqu'à ce que l'ensemble des points ait été lu.

Afin de pouvoir définir plusieurs surfaces, manipulables par la suite, l'ensemble des surfaces est stocké dans un tableau à trois indices qui indiquent le numéro de la surface, l'indicateur et les coordonnées de ses points.

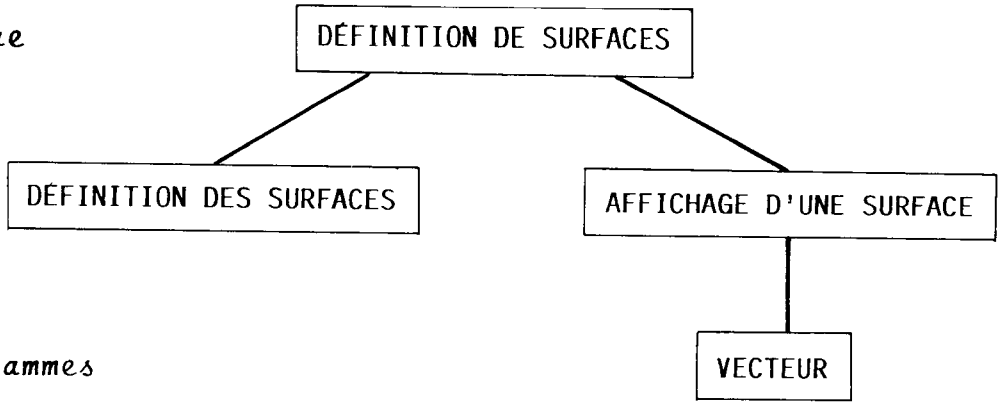
EXEMPLE D'EXECUTION

Puisque nous allons définir une surface lors de l'explication de la méthode de mise en oeuvre, nous allons la tracer accompagnée d'une autre surface définie de la même façon. On peut voir le résultat de l'exécution page suivante.

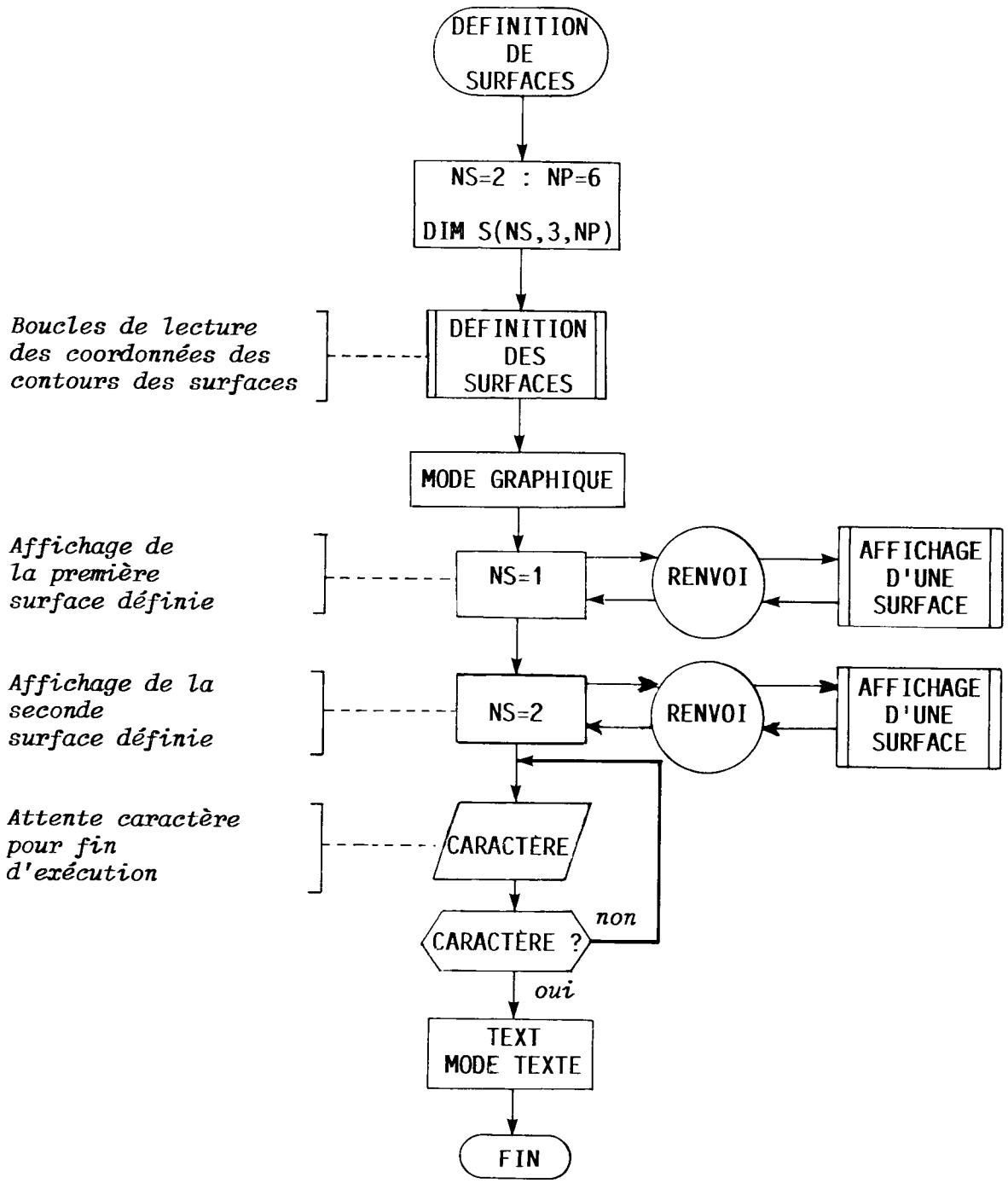
LE PROGRAMME

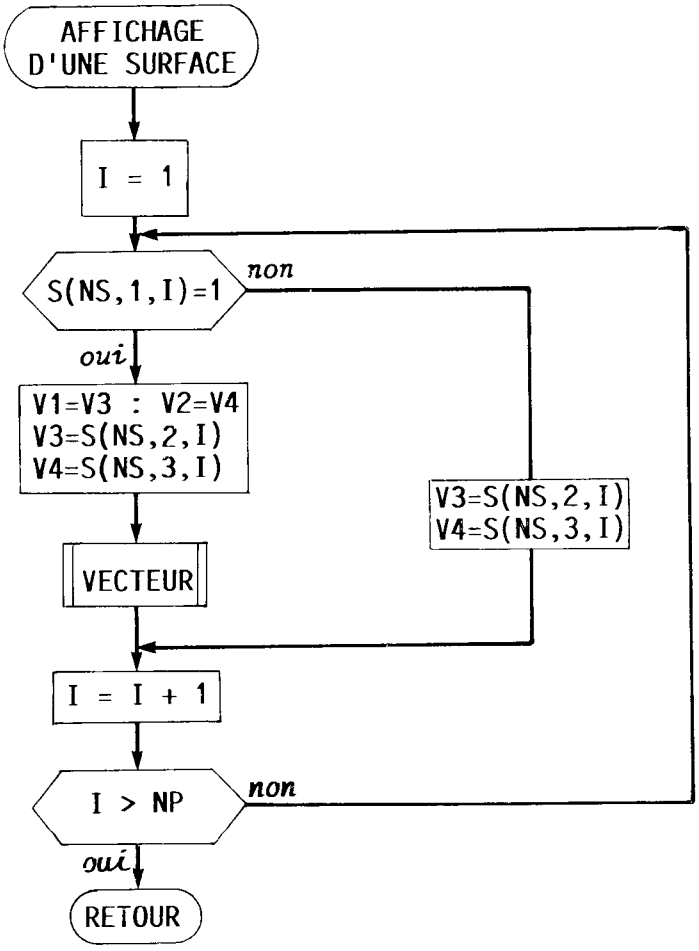
Traitement

Structure



Organigrammes




AFFICHAGE
D'UNE SURFACE
I = 1
S(NS,1,I)=1
non
oui
V1=V3 : V2=V4
V3=S(NS,2,I)
V4=S(NS,3,I)
VECTEUR
V3=S(NS,2,I)
V4=S(NS,3,I)
I = I + 1
I > NP
non
oui
RETOUR

Variables

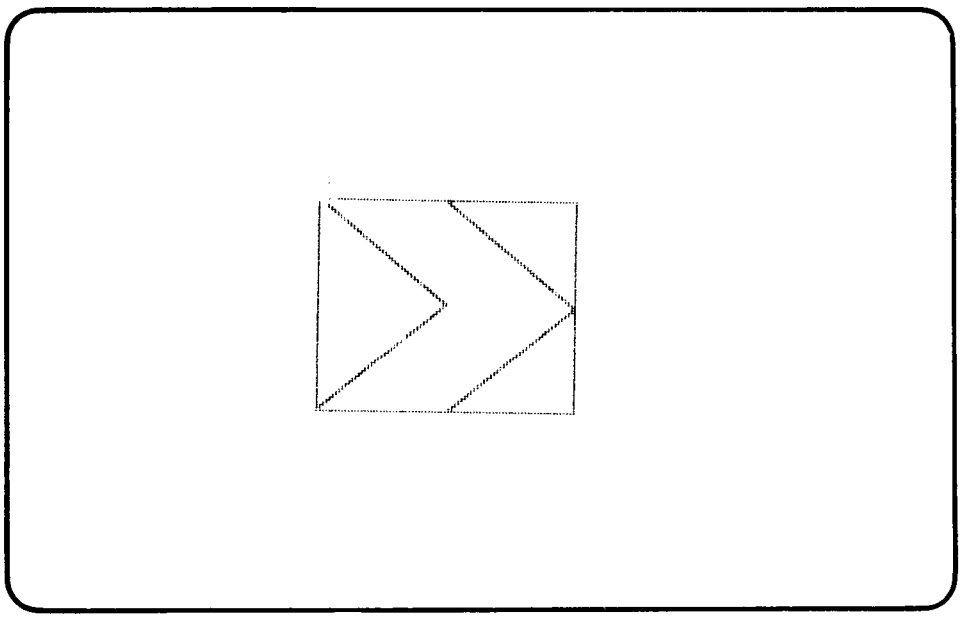
DEFINITION DE SURFACES	:	Programme principal
NS	:	Nombre total de surfaces et numéro de surface
NP	:	Nombre maximum de points
S (NS,3,NP)	:	Tableau des contours
R$	:	Caractère de fin d'exécution
VECTEUR	:	Tracé des vecteurs
V1,V2	:	Coordonnées du premier point du vecteur à tracer
V3,V4	:	Coordonnées du second point du vecteur à tracer
AFFICHAGE D'UNE SURFACE	:	Trace le contour d'une surface
S (NS,3,NP)	:	Tableau des points de la surface à tracer
V1...V4	:	Coordonnées transférées au sous-programme VECTEUR
DEFINITION DES SURFACES	:	Lecture des DATA définissant les surfaces
NS	:	Nombre total de surfaces à lire
I,J,K	:	Indices de lecture
NP	:	Nombre de points à lire pour chaque surface

Particularité

On peut noter qu'étant donné l'utilisation de l'indicateur d'affichage, il est possible de tracer plusieurs contours pour une même définition de surface. Cependant un tel usage peut poser quelques problèmes dans certains cas, comme nous le verrons par ailleurs.

```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM DEFINITION DE SURFACES
1004  REM ------------------------------------------------------------
1005  REM NOMBRE DE SURFACES
1010 NS = 2
1015  REM NOMBRE MAXIMUM DE POINTS
1020 NP = 6
1025  REM DECLARATION DES TABLEAUX
1030  DIM S(NS,3,NP)
1035  REM DEFINITION DES SURFACES
1040  GOSUB 6000: REM * DEFINITION DES SURFACES *
1045  REM  PAGE 1 GRAPHIQUE
1050  HGR : POKE  - 16302,0: HCOLOR= 3
1055  REM AFFICHAGE DES SURFACES
1060 NS = 1: GOSUB 5000: REM * AFFICHAGE D'UNE SURFACE *

1070 NS = 2: GOSUB 5000: REM * AFFICHAGE D'UNE SURFACE *

1080  GET R$: IF R$ = "" THEN 1080
1090  HOME : TEXT
1100  END : REM ------------------------------------------------------
1110  REM
1120  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ---------------------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------------
5002  REM AFFICHAGE D'UNE SURFACE
5004  REM ------------------------------------------------------------
5010  FOR I = 1 TO NP
5020  IF S(NS,1,I) = 1 THEN 5050
5030 V3 = S(NS,2,I):V4 = S(NS,3,I)
5040  GOTO 5080
5050 V1 = V3:V2 = V4
5060 V3 = S(NS,2,I):V4 = S(NS,3,I)
5070  GOSUB 4000: REM * VECTEUR *
5080  NEXT I
5090  RETURN : REM ---------------------------------------------------
5100  REM
5110  REM
```

```
6000  REM ------------------------------------------------------------
6002  REM DEFINITION DES SURFACES
6004  REM ------------------------------------------------------------
6010  FOR I = 1 TO NS
6020  READ NP
6030  FOR J = 1 TO NP
6040  FOR K = 1 TO 3
6050  READ S(I,K,J)
6060  NEXT K
6070  NEXT J
6080  NEXT I
6085  REM PREMIERE SURFACE
6090  DATA  6
6100  DATA  0,0,0
6110  DATA  1,40,40
6120  DATA  1, 0,80
6130  DATA  1,80,80
6140  DATA  1,80,0
6150  DATA  1,0,0
6155  REM SECONDE SURFACE
6160  DATA  6
6170  DATA  0,0,0
6180  DATA  1,0,80
6190  DATA  1,40,80
6200  DATA  1,80,40
6210  DATA  1,40,0
6220  DATA  1,0,0
6230  RETURN : REM ----------------------------------------------------
6240  REM
```



TRANSLATION DE SURFACES

La translation est une transformation que nous connaissons maintenant très bien puisque nous l'avons déjà utilisée pour le point et le polygone. Ici, **c'est la généralisation au cas des surfaces que nous allons envisager.** La translation est un outil essentiel à la constitution des scènes que nous souhaitons réaliser car elle nous permettra le placement des surfaces.

HYPOTHESE

Nous allons conserver le mode de définition et de tracé que nous avons mis en oeuvre précédemment. Nous allons donc utiliser les sous-programmes ***DEFINITION D'UNE SURFACE***, ***AFFICHAGE D'UNE SURFACE*** et ***VECTEUR*** auxquels nous allons ajouter le sous-programme ***TRANSLATION D'UNE SURFACE.***

MODELE

Sans détailler l'écriture du sous-programme puisque nous l'avons déjà fait lors du chapitre précédent, notons simplement que lorsque la transformation est effectuée, on trouve la surface résultante dans le tableau S d'indice **NS = Ø.** Ce transfert est réalisé afin de conserver la définition originelle de la surface transformée par une éventuelle utilisation ultérieure.

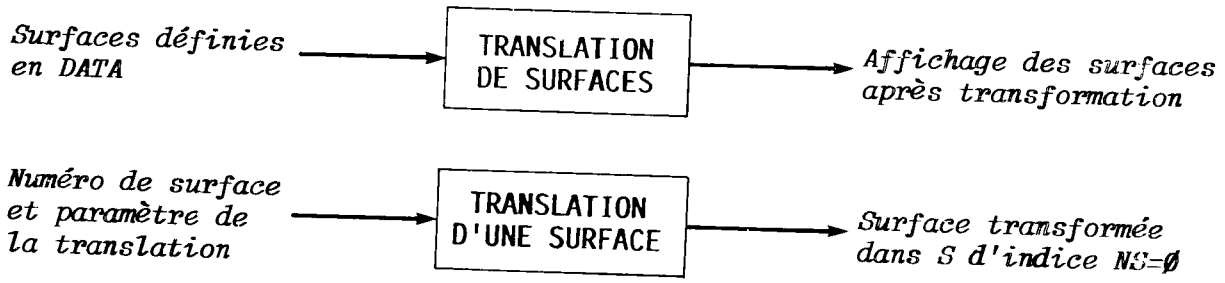
EXEMPLE D'EXECUTION

Le programme exemple montre la translation des deux surfaces définies lors du chapitre précédent. Comme on peut le constater

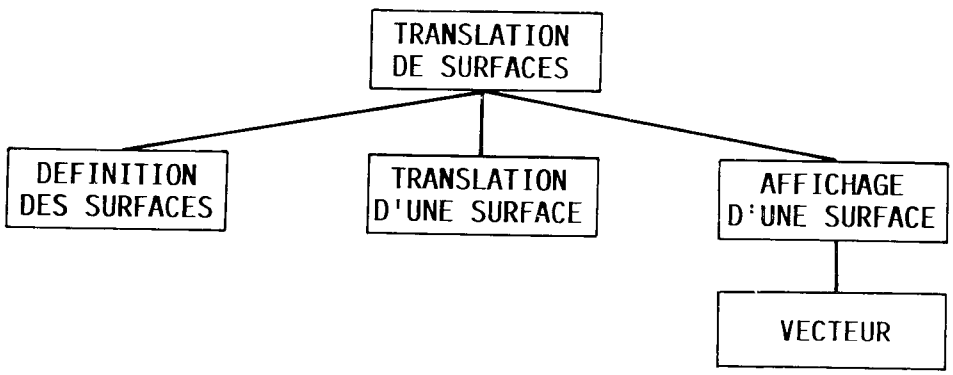
page 148, cette transformation permet une réelle disposition à volonté des surfaces définies.

LE PROGRAMME

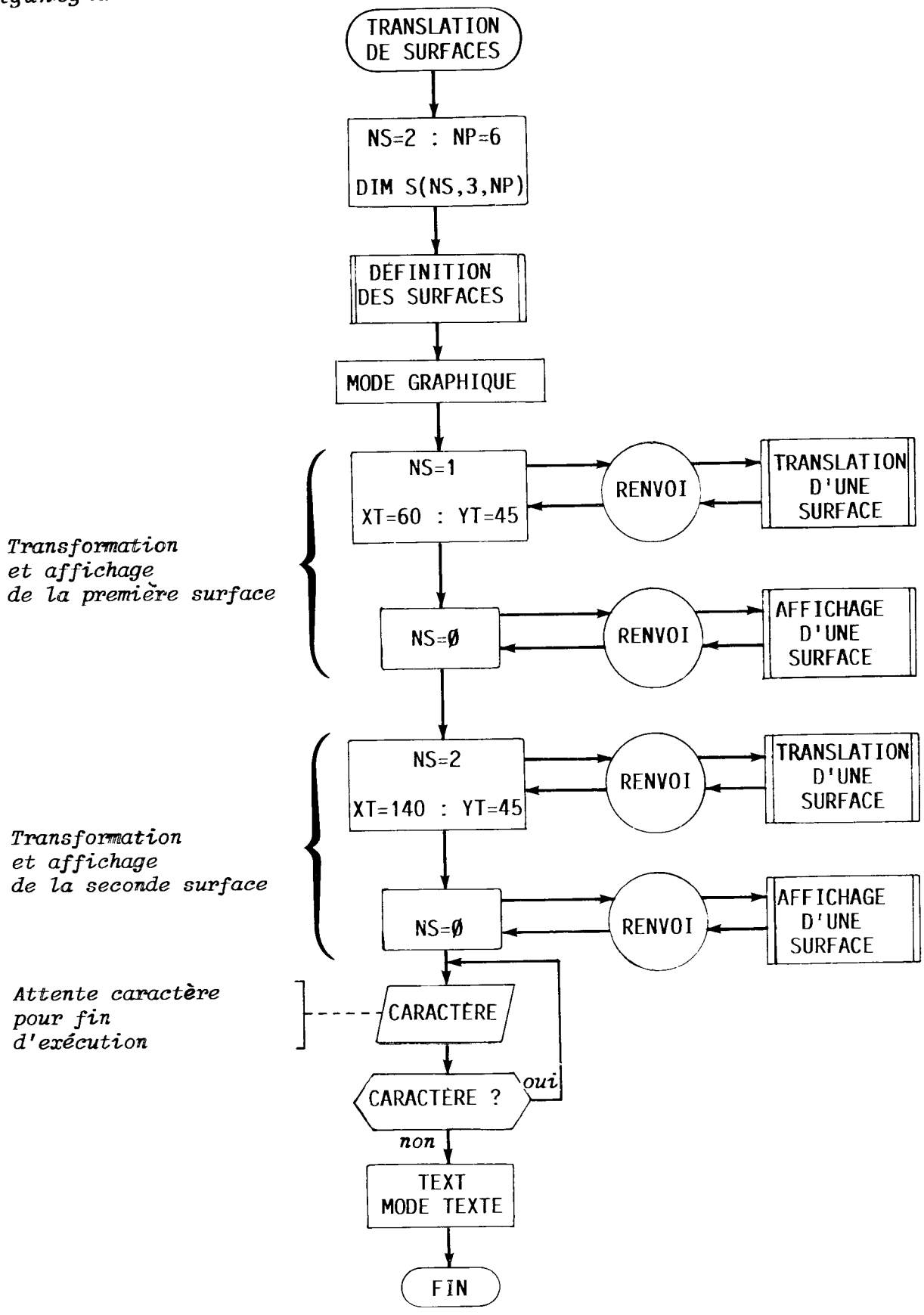
Traitement

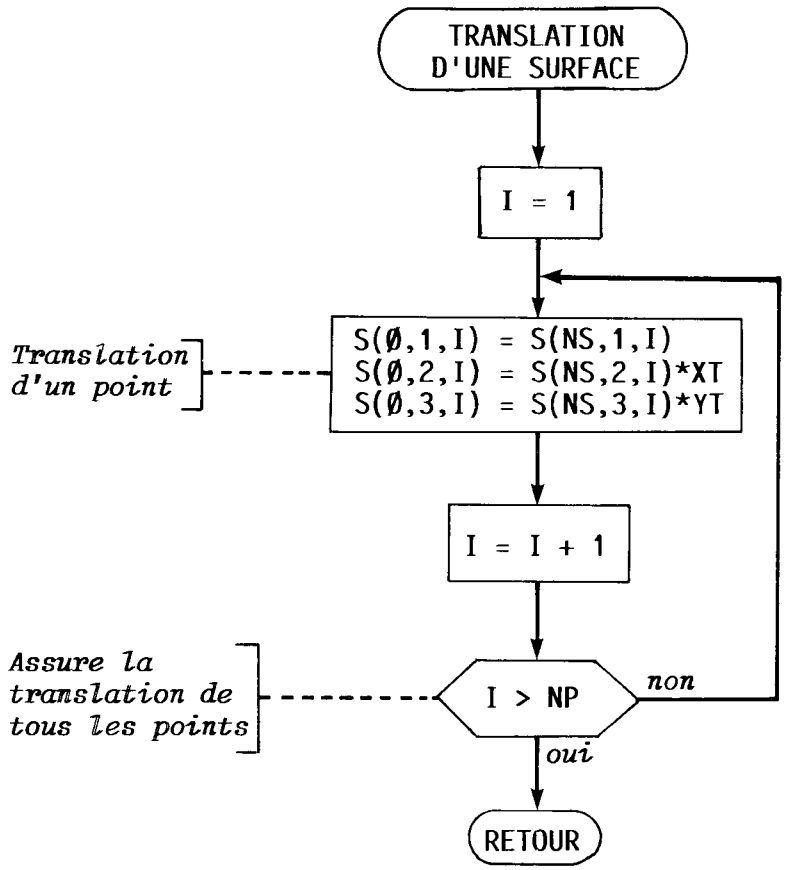


Structure



Organigrammes





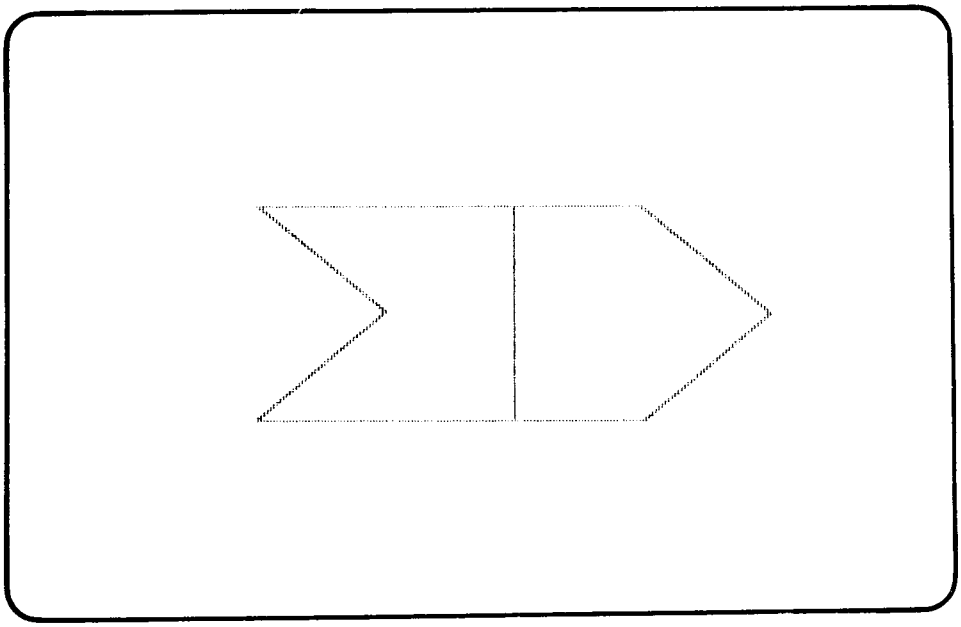
Variables

TRANSLATION DE SURFACES	:	Programme principal
NS	:	Nombre total de surfaces ou numéro de surface
NP	:	Nombre maximum de points
S (NS,3,NP)	:	Tableau des contours
XT,YT	:	Paramètres de translation
R$	:	Caractère de fin d'exécution
VECTEUR	:	Tracé de vecteurs
V1, V2	:	Coordonnées du premier point de vecteur à tracer
V3,V4	:	Coordonnées du second point de vecteur à tracer
TRANSLATION D'UNE SURFACE	:	Transformation des paramètres NS, XT et YT
NS	:	Numéro de surface à transformer
XT,YT	:	Paramètres de la translation
S (NS,3,NP)	:	Tableau des points du contour
I	:	Indice de boucle

AFFICHAGE D'UNE SURFACE	:	Trace le contour d'une surface
S (NS,3,NP)	:	Tableau des points de la surface à tracer
V1...V4	:	Coordonnées transférées au sous-programme VECTEUR
DEFINITION DES SURFACES	:	Lecture des DATA définissant les surfaces.
NS	:	Nombre total de surfaces
I,J,K	:	Indices de lecture
NP	:	Nombre de points à lire pour chaque surface

```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRANSLATION DE SURFACES
1004  REM ------------------------------------------------------------
1005  REM NOMBRE DE SURFACES
1010 NS = 2
1015  REM NOMBRE MAXIMUM DE POINTS
1020 NP = 6
1025  REM DECLARATION DES TABLEAUX
1030  DIM S(NS,3,NP)
1035  REM DEFINITION DES SURFACES
1040  GOSUB 7000: REM * DEFINITION DES SURFACES *
1045  REM  PAGE 1 GRAPHIQUE
1050  HGR : POKE  - 16302,0: HCOLOR= 3
1055  REM AFFICHAGE DES SURFACES
1060 NS = 1:XT = 60:YT = 45
1070  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1080 NS = 0: GOSUB 6000: REM * AFFICHAGE D'UNE SURFACE *

1090 NS = 2:XT = 140:YT = 45
1100  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1110 NS = 0: GOSUB 6000: REM * AFFICHAGE D'UNE SURFACE *

1120  GET R$: IF R$ = "" THEN 1120
1130  HOME : TEXT
1140  END : REM ------------------------------------------------------
1150  REM
1160  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ----------------------------------------------------
4050  REM
4060  REM
```

```
5000  REM ------------------------------------------------------------
5002  REM TRANSLATION D'UNE SURFACE
5004  REM ------------------------------------------------------------
5010  FOR I = 1 TO NP
5020 S(0,1,I) = S(NS,1,I)
5030 S(0,2,I) = S(NS,2,I) + XT
5040 S(0,3,I) = S(NS,3,I) + YT
5050  NEXT I
5060  RETURN : REM ------------------------------------------------------
5070  REM
5080  REM
6000  REM ------------------------------------------------------------
6002  REM AFFICHAGE D'UNE SURFACE
6004  REM ------------------------------------------------------------
6010  FOR I = 1 TO NP
6020  IF S(NS,1,I) = 1 THEN 6050
6030 V3 = S(NS,2,I):V4 = S(NS,3,I)
6040  GOTO 6080
6050 V1 = V3:V2 = V4
6060 V3 = S(NS,2,I):V4 = S(NS,3,I)
6070  GOSUB 4000: REM * VECTEUR *
6080  NEXT I
6090  RETURN : REM ------------------------------------------------------
6100  REM
6110  REM
7000  REM ------------------------------------------------------------
7002  REM DEFINITION DES SURFACES
7004  REM ------------------------------------------------------------
7010  FOR I = 1 TO NS
7020  READ NP
7030  FOR J = 1 TO NP
7040  FOR K = 1 TO 3
7050  READ S(I,K,J)
7060  NEXT K
7070  NEXT J
7080  NEXT I
7085  REM PREMIERE SURFACE
7090  DATA  6
7100  DATA  0,0,0
7110  DATA  1,40,40
7120  DATA  1,  0,80
7130  DATA  1,80,80
7140  DATA  1,80,0
7150  DATA  1,0,0
7155  REM SECONDE SURFACE
7160  DATA  6
7170  DATA  0,0,0
7180  DATA  1,0,80
7190  DATA  1,40,80
7200  DATA  1,80,40
7210  DATA  1,40,0
7220  DATA  1,0,0
7230  RETURN : REM ------------------------------------------------------
7240  REM
```



ROTATION DE SURFACES

Comme la translation, nous connaissons bien la rotation puisqu'elle aussi, nous l'avons décomposée dans le cas de la transformation d'un point ou d'un polygone. **Nous allons généraliser la rotation au cas des surfaces**, sans toutefois utiliser toutes ses possibilités comme nous le verrons par la suite.

HYPOTHESE

Nous allons écrire un sous-programme permettant la mise en oeuvre de rotations de surfaces que nous allons ajouter au programme ***TRANSLATION de SURFACES***. Ceci nous permet de conserver le bénéfice des routines réalisées lors de précédents chapitres.

MODELE

Nous n'allons considérer que la rotation de centre égal à l'origine du repère lié à l'écran, cela parce que les surfaces sont toutes définies à partir du point de coordonnées (Ø,Ø) et que notre seul but est de les placer dans un plan. En conséquence, nous choisirons d'orienter la surface par rotation puis, de la placer par translation. Toutes les possibilités de positionnement sont ainsi couvertes sans qu'il soit nécessaire de préciser de centre de rotation.

Notons que si nous envisagions d'utiliser le sous-programme ***ROTATION*** pour réaliser une éventuelle animation, nous aurions conservé la possibilité de fixer le centre de rotation car certains calculs auraient été plus rapides. Dans un tel cas, le lecteur n'aurait aucune difficulté à modifier le sous-programme

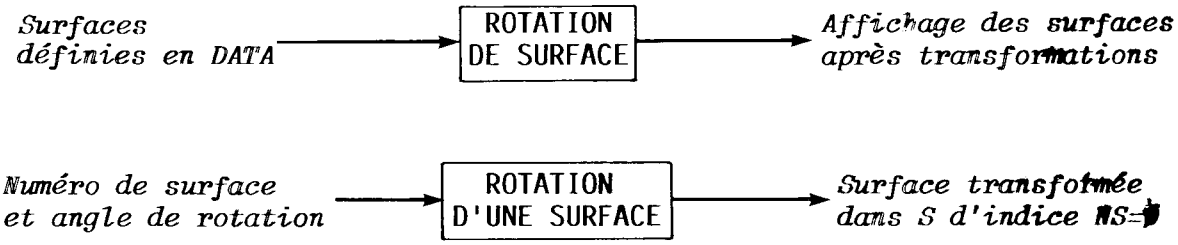
puisque nous avons vu la rotation de centre fixée par l'utilisateur dans un chapitre précédent.

EXEMPLE D'EXECUTION

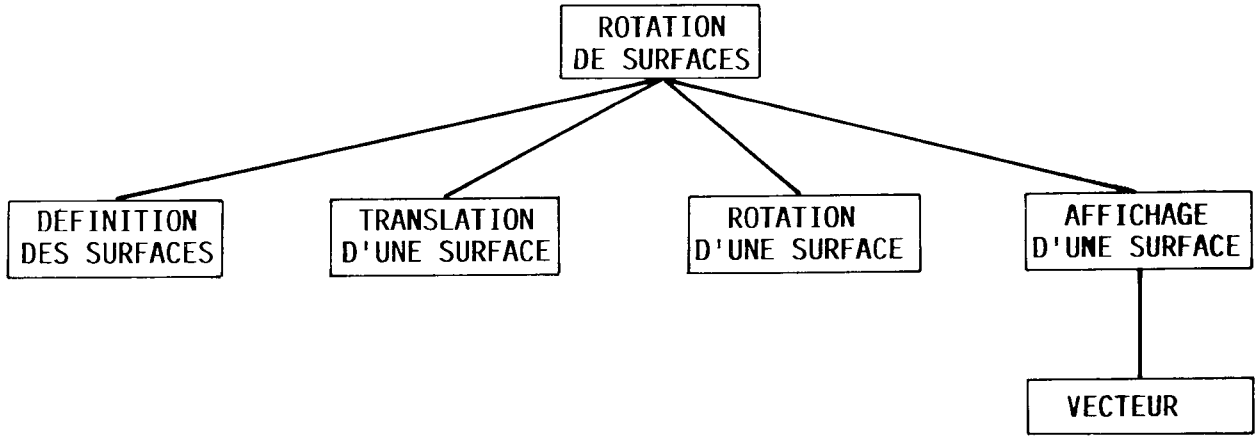
En utilisant, une fois encore, les surfaces définies lors du chapitre **DEFINITION DE SURFACES** , le programme des pages suivantes construit une nouvelle scène utilisant les transformations ***TRANSLATION-ROTATION*** et ***ROTATION-TRANSLATION.*** Ce sont les deux possibilités de transformations composées qui sont mises en oeuvre et dont on peut voir le résultat d'exécution page 155.

LE PROGRAMME

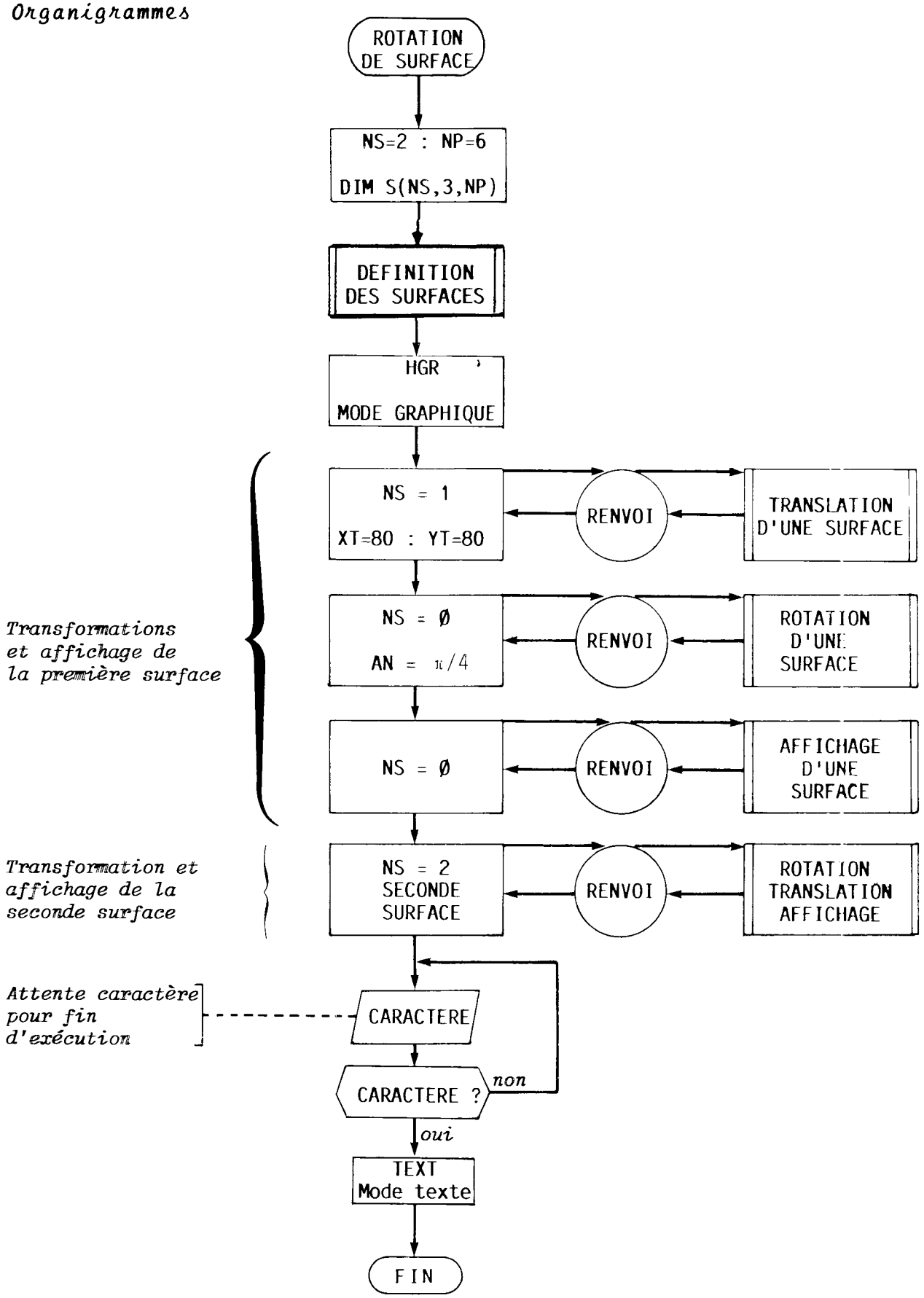
Traitement

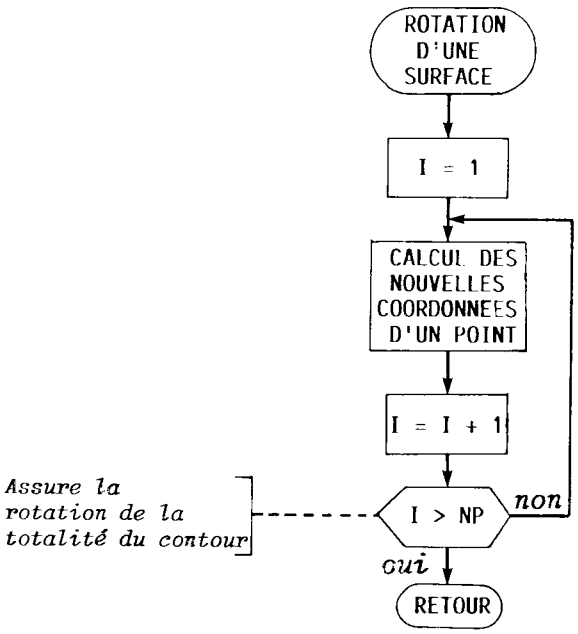


Structure



Organigrammes



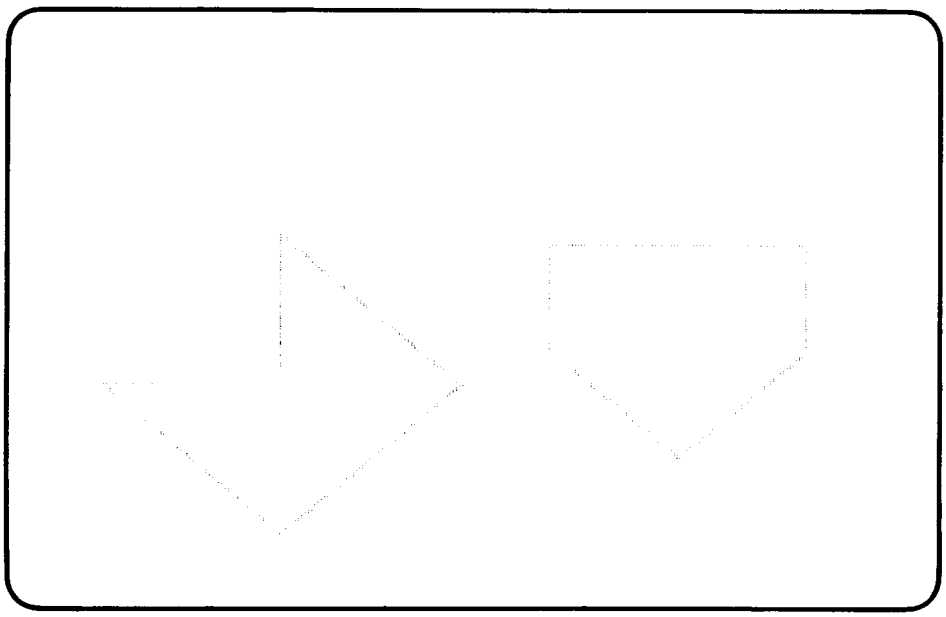


Variables

ROTATION DE SURFACES	:	Programme principal
NS	:	Nombre total de surfaces ou numéro de surface
NP	:	Nombre maximum de points
S (NS,3,NP)	:	Tableau des contours
XT,YT	:	Paramètres de translation
AN	:	Angle de rotation
R$	:	Caractère de fin d'exécution
VECTEUR	:	Tracé de vecteurs
V1,V2	:	Coordonnées du premier point de vecteur à tracer
V3,V4	:	Coordonnées de second point du vecteur à tracer
TRANSLATION D'UNE SURFACE	:	Transformation de paramètres **NS, XT** et **YT.**
NS	:	Numéro de surface
XT,YT	:	Paramètres de la translation
S (NS,3,NP)	:	Tableau des points du contour
I	:	Indice de boucle
AFFICHAGE D'UNE SURFACE	:	Trace le contour d'une surface
S (NS,3,NP)	:	Tableau des points de la surface à tracer
V1...V4	:	Coordonnées transférées au sous-programme VECTEUR
DEFINITION DES SURFACES	:	Lecture des DATA définissant les surfaces
NS	:	Nombre total de surfaces
I,J,K	:	Indices de lecture
NP	:	Nombre de points à lire pour chaque surface

```
$LIST1000-

1000  REM ------------------------------------------------------
1002  REM ROTATION DE SURFACES
1004  REM ------------------------------------------------------
1005  REM NOMBRE DE SURFACES
1010 NS = 2
1015  REM NOMBRE MAXIMUM DE POINTS
1020 NP = 6
1025  REM DECLARATION DES TABLEAUX
1030  DIM S(NS,3,NP)
1035  REM DEFINITION DES SURFACES
1040  GOSUB 8000: REM * DEFINITION DES SURFACES *
1045  REM  PAGE 1 GRAPHIQUE
1050  HGR : POKE  - 16302,0: HCOLOR= 3
1055  REM AFFICHAGE DES SURFACES
1060 NS = 1:XT = 80:YT = 0
1070  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1080 NS = 0:AN = 3.14159 / 4
1090  GOSUB 6000: REM * ROTATION D'UNE SURFACE *
1100 NS = 0: GOSUB 7000: REM * AFFICHAGE D'UNE SURFACE *

1110 NS = 2:AN = 3.14159 / 2
1120  GOSUB 6000: REM * ROTATION D'UNE SURFACE *
1130 NS = 0:XT = 220:YT = 60
1140  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1150 NS = 0: GOSUB 7000: REM * AFFICHAGE D'UNE SURFACE *

1160  GET R$: IF R$ = "" THEN 1160
1170  HOME : TEXT
1180  END : REM ------------------------------------------------
1190  REM
1200  REM
4000  REM ------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ----------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------------
5002  REM TRANSLATION D'UNE SURFACE
5004  REM ------------------------------------------------------
5010  FOR I = 1 TO NP
5020 S(0,1,I) = S(NS,1,I)
5030 S(0,2,I) = S(NS,2,I) + XT
5040 S(0,3,I) = S(NS,3,I) + YT
5050  NEXT I
5060  RETURN : REM ----------------------------------------
5070  REM
5080  REM
```

```
6000  REM ------------------------------------------------------
6002  REM ROTATION D'UNE SURFACE
6004  REM ------------------------------------------------------
6010  FOR I = 1 TO NP
6020 XP = S(NS,2,I) *  COS (AN) - S(NS,3,I) *  SIN (AN)
6030 YP = S(NS,2,I) *  SIN (AN) + S(NS,3,I) *  COS (AN)
6040 S(0,2,I) = XP:S(0,3,I) = YP
6050 S(0,1,I) = S(NS,1,I)
6060  NEXT I
6070  RETURN : REM ----------------------------------------------
6080  REM
6090  REM
7000  REM ------------------------------------------------------
7002  REM AFFICHAGE D'UNE SURFACE
7004  REM ------------------------------------------------------
7010  FOR I = 1 TO NP
7020  IF S(NS,1,I) = 1 THEN 7050
7030 V3 = S(NS,2,I):V4 = S(NS,3,I)
7040  GOTO 7080
7050 V1 = V3:V2 = V4
7060 V3 = S(NS,2,I):V4 = S(NS,3,I)
7070  GOSUB 4000: REM * VECTEUR *
7080  NEXT I
7090  RETURN : REM ----------------------------------------------
7100  REM
7110  REM
8000  REM ------------------------------------------------------
8002  REM DEFINITION DES SURFACES
8004  REM ------------------------------------------------------
8010  FOR I = 1 TO NS
8020  READ NP
8030  FOR J = 1 TO NP
8040  FOR K = 1 TO 3
8050  READ S(I,K,J)
8060  NEXT K
8070  NEXT J
8080  NEXT I
8085  REM PREMIERE SURFACE
8090  DATA  6
8100  DATA  0,0,0
8110  DATA  1,40,40
8120  DATA  1, 0,80
8130  DATA  1,80,80
8140  DATA  1,80,0
8150  DATA  1,0,0
8155  REM SECONDE SURFACE
8160  DATA  6
8170  DATA  0,0,0
8180  DATA  1,0,80
8190  DATA  1,40,80
8200  DATA  1,80,40
8210  DATA  1,40,0
8220  DATA  1,0,0
8230  RETURN : REM ----------------------------------------------
8240  REM
```



AGRANDISSEMENT DE SURFACES

Dans ce chapitre, nous allons aborder une nouvelle transformation qui va vous permettre de modifier à volonté la taille de chacune des surfaces composant une scène.

HYPOTHESE

Comme dans le cas de la translation et de la rotation, nous allons écrire un sous-programme indépendant des surfaces à transformer. Ce sous-programme va donc effectuer des agrandissements de surfaces sans créer d'interférence sur un programme du type de celui du chapitre précédent auquel nous allons ajouter cette nouvelle fonction.

MODELE

L'augmentation ou la diminution de la taille d'une surface est obtenue simplement par multiplication des coordonnées du contour par un coefficient. Ainsi, si **P1...Pn** sont les n points du contour et **F** le facteur d'agrandissement, on aura, sous forme matricielle, et pour chaque point, la transformation suivante à effectuer :

$$\begin{bmatrix} X'i \\ Y'i \end{bmatrix} = \begin{bmatrix} F & 0 \\ 0 & F \end{bmatrix} \begin{bmatrix} Xi \\ Yi \end{bmatrix}$$

d'où les coordonnées transformées du point **Pi** du contour.

$$Pi = \begin{bmatrix} X'i \\ Y'i \end{bmatrix} = \begin{bmatrix} F.Xi \\ F.Yi \end{bmatrix}$$

Lorsque tous les points du contour ont été transformés par un facteur F non nul, la taille de la surface a été modifiée.

Notons que lorsque la surface subit la transformation alors qu'aucun de ces points n'est à l'origine de l'écran, elle subit également une translation. Il faut dans un tel cas lui appliquer une translation inverse pour ramener la surface à sa position initiale. Afin d'éviter que la surface ne sorte de l'écran, nous procéderons généralement à l'agrandissement avant toute autre transformation.

Notons encore la possibilité d'effectuer des distorsions sur les surfaces en utilisant deux facteurs d'agrandissement, l'un suivant l'axe-**X** et l'autre suivant l'axe-**Y**. Cette possibilité n'est pas mise en oeuvre dans ce chapitre.

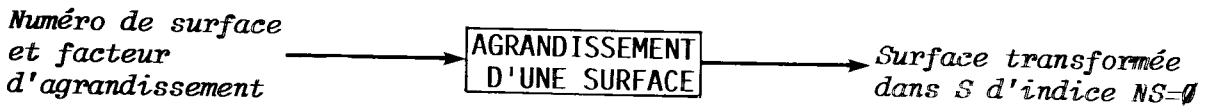
EXEMPLE D'UTILISATION

Les surfaces que nous avons transformées par la translation et la rotation étaient définies dans leur taille d'affichage. Cette fois, elles sont toujours définies à l'origine du repère lié à l'écran, mais la définition est effectuée dans le plus petit carré contenant les surfaces. La taille d'affichage est obtenue ensuite par agrandissement des contours aux dimensions voulues. Les surfaces sont enfin positionnées par les transformations des chapitres précédents.

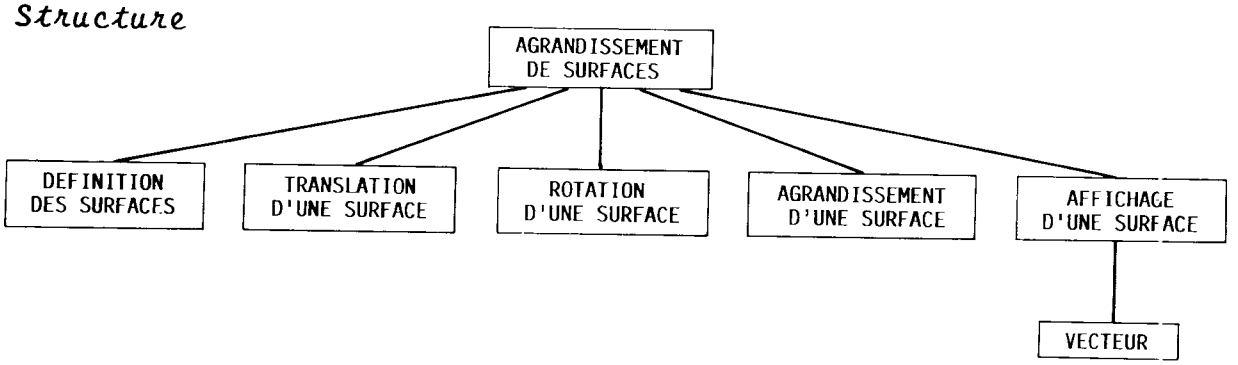
Le programme exemple transforme deux surfaces comme on peut l'observer page 163.

LE PROGRAMME

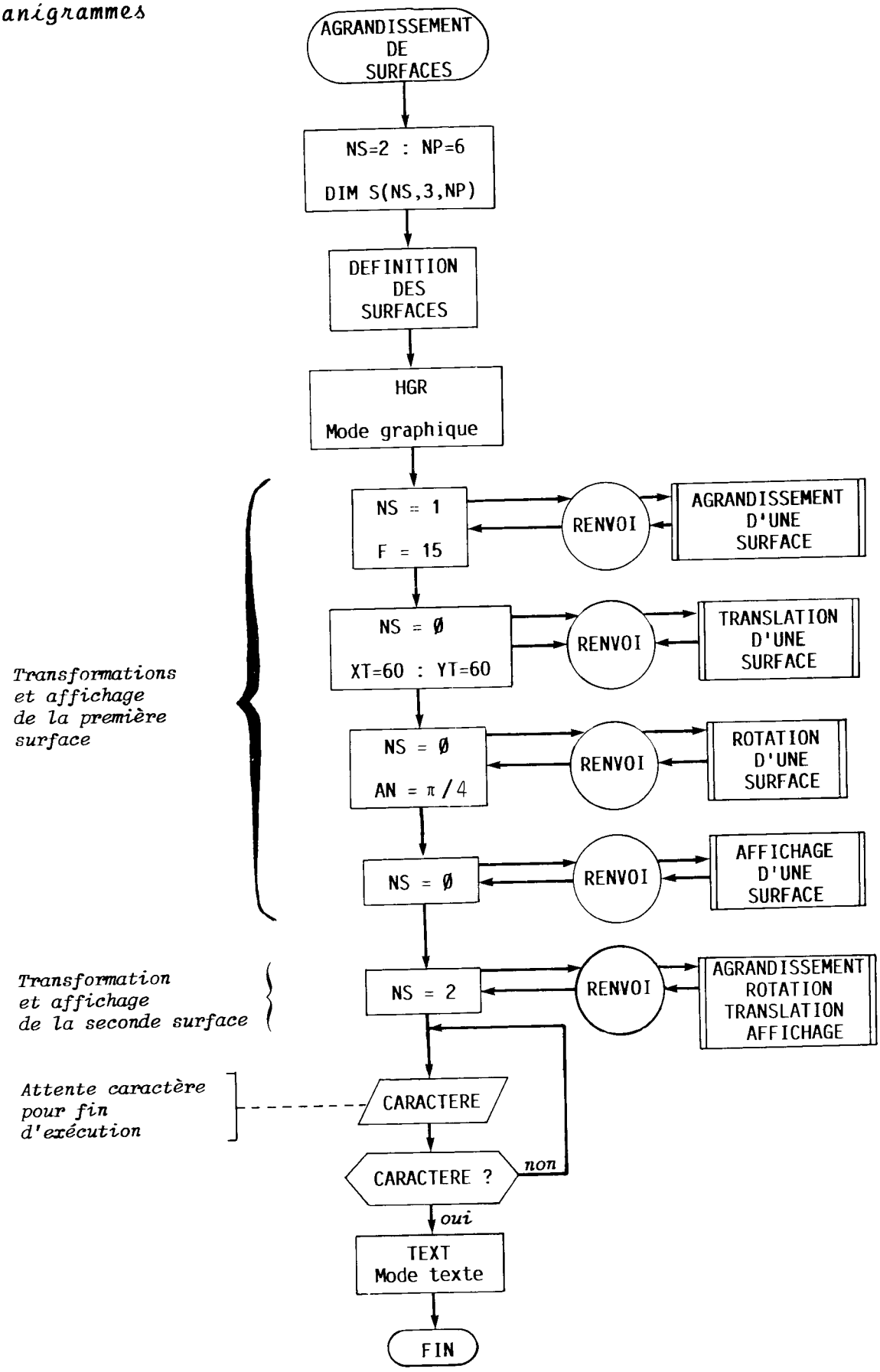
Traitement

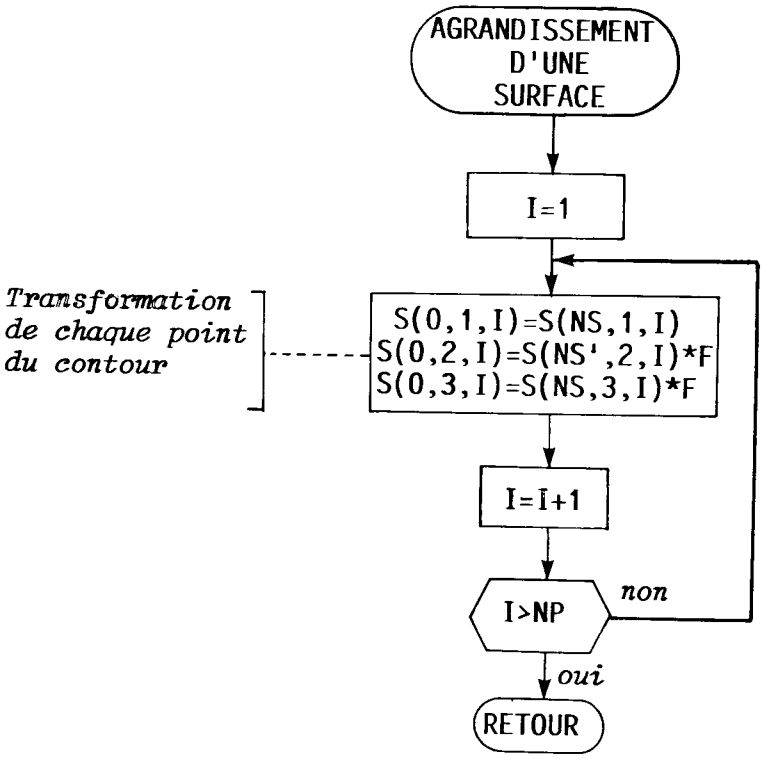


Structure



Organigrammes




AGRANDISSEMENT D'UNE SURFACE
I=1
Transformation de chaque point du contour
S(0,1,I)=S(NS,1,I)
S(0,2,I)=S(NS',2,I)*F
S(0,3,I)=S(NS,3,I)*F
I=I+1
I>NP
non
oui
RETOUR

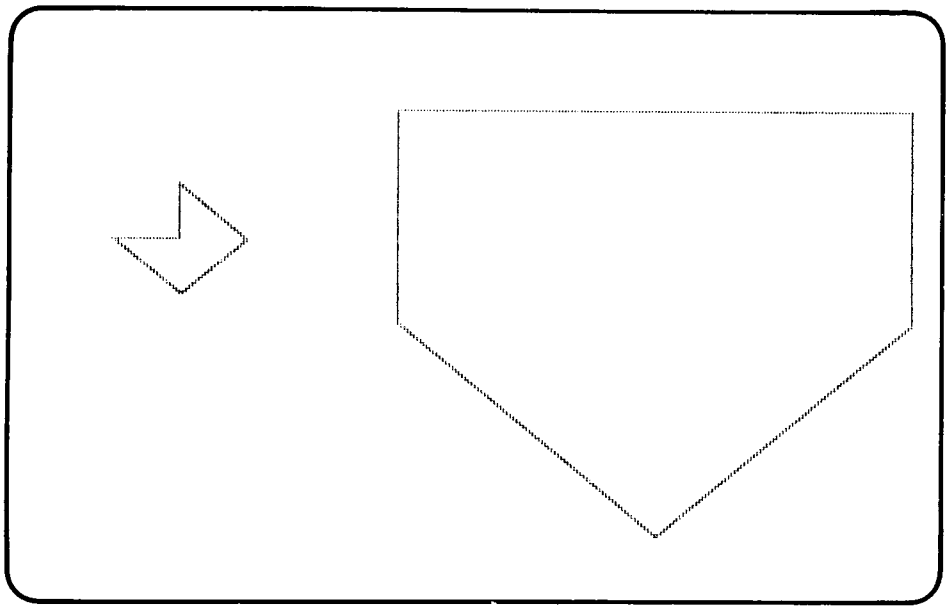
Variables

AGRANDISSEMENT DE SURFACES	:	Programme principal
NS		Nombre total de surfaces ou numéro de surface
NP	:	Nombre maximum de points
S (NS,3,NP)	:	Tableau des contours
F	:	Facteur d'agrandissement
XT,YT	:	Paramètres de translation
AN	:	Angle de rotation
R$	:	Caractère de fin d'exécution
VECTEUR	:	Tracé de vecteurs
V1,V2	:	Coordonnées du premier point du vecteur à tracer
V3,V4	:	Coordonnées du second point du vecteur à tracer
TRANSLATION D'UNE SURFACE	:	Transformation de paramètres **NS, XT et YT**
NS	:	Numéro de surface
XT,YT	:	Paramètres de la translation
S (NS,3,NP)	:	Tableau des points du contour
I	:	Indice de boucle
ROTATION D'UNE SURFACE	:	Transformation de paramètres **AN** et **NS**
NS	:	Numéro de surface
XP,YP	:	Coordonnées transformées des points
S (NS,3,NP)	:	Tableau des points du contour
AN	:	Angle de rotation
I	:	Indice de boucle
AGRANDISSEMENT D'UNE SURFACE	:	Fait varier la taille d'une surface
F	:	Facteur d'agrandissement
S (NS,3,NP)	:	Tableau des points du contour
I	:	Indice de boucle
AFFICHAGE D'UNE SURFACE	:	Trace le contour d'une surface
S (NS,3,NP)	:	Tableau des points de la surface à tracer
V1...V4	:	Coordonnées transférées au sous-programme VECTEUR
DEFINITION DES SURFACES	:	Lecture des DATA définissant les surfaces
NS	:	Nombre total de surfaces
I,J,K	:	Indices de lecture
NP	:	Nombre de points à lire pour chaque surface

```
$LIST1000-

1000  REM ------------------------------------------------------------
1002  REM AGRANDISSEMENT DE SURFACES
1004  REM ------------------------------------------------------------
1005  REM NOMBRE DE SURFACES
1010 NS = 2
1015  REM NOMBRE MAXIMUM DE POINTS
1020 NP = 6
1025  REM DECLARATION DES TABLEAUX
1030  DIM S(NS,3,NP)
1035  REM DEFINITION DES SURFACES
1040  GOSUB 9000: REM * DEFINITION DES SURFACES *
1045  REM  PAGE 1 GRAPHIQUE
1050  HGR : POKE  - 16302,0: HCOLOR= 3
1055  REM AFFICHAGE DES SURFACES
1060 NS = 1:F = 15
1070  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURF
1080 NS = 0:XT = 60:YT = 0
1090  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE
1100 NS = 0:AN = 3.14159 / 4
1110  GOSUB 6000: REM * ROTATION D'UNE SURFACE *
1120 NS = 0: GOSUB 8000: REM * AFFICHAGE D'UNE SU

1130 NS = 2:F = 80
1140  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURF
1150 NS = 0:AN = 3.14159 / 2
1160  GOSUB 6000: REM * ROTATION D'UNE SURFACE *
1170 NS = 0:XT = 270:YT = 15
1180  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE
1190 NS = 0: GOSUB 8000: REM * AFFICHAGE D'UNE SU

1200  GET R$: IF R$ = "" THEN 1200
1210  HOME : TEXT
1220  END : REM ------------------------------------------------------
1230  REM
1240  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ----------------------------------------------------
4050  REM
4060  REM
```

```
5000  REM ------------------------------------------------------------
5002  REM TRANSLATION D'UNE SURFACE
5004  REM ------------------------------------------------------------
5010  FOR I = 1 TO NP
5020 S(0,1,I) = S(NS,1,I)
5030 S(0,2,I) = S(NS,2,I) + XT
5040 S(0,3,I) = S(NS,3,I) + YT
5050  NEXT I
5060  RETURN : REM ------------------------------------------------------
5070  REM
5080  REM
6000  REM ------------------------------------------------------------
6002  REM ROTATION D'UNE SURFACE
6004  REM ------------------------------------------------------------
6010  FOR I = 1 TO NP
6020 XP = S(NS,2,I) *  COS (AN) - S(NS,3,I) *  SIN (AN)
6030 YP = S(NS,2,I) *  SIN (AN) + S(NS,3,I) *  COS (AN)
6040 S(0,2,I) = XP:S(0,3,I) = YP
6050 S(0,1,I) = S(NS,1,I)
6060  NEXT I
6070  RETURN : REM ------------------------------------------------------
6080  REM
6090  REM
7000  REM ------------------------------------------------------------
7002  REM AGRANDISSEMENT D'UNE SURFACE
7004  REM ------------------------------------------------------------
7010  FOR I = 1 TO NP
7020 S(0,1,I) = S(NS,1,I)
7030 S(0,2,I) = S(NS,2,I) * F
7040 S(0,3,I) = S(NS,3,I) * F
7050  NEXT I
7060  RETURN : REM ------------------------------------------------------
7070  REM
7080  REM
8000  REM ------------------------------------------------------------
8002  REM AFFICHAGE D'UNE SURFACE
8004  REM ------------------------------------------------------------
8010  FOR I = 1 TO NP
8020  IF S(NS,1,I) = 1 THEN 8050
8030 V3 = S(NS,2,I):V4 = S(NS,3,I)
8040  GOTO 8080
8050 V1 = V3:V2 = V4
8060 V3 = S(NS,2,I):V4 = S(NS,3,I)
8070  GOSUB 4000: REM * VECTEUR *
8080  NEXT I
8090  RETURN : REM ------------------------------------------------------
8100  REM
8110  REM
```

```
9000  REM ------------------------------------------------------------
9002  REM DEFINITION DES SURFACES
9004  REM ------------------------------------------------------------
9010  FOR I = 1 TO NS
9020  READ NP
9030  FOR J = 1 TO NP
9040  FOR K = 1 TO 3
9050  READ S(I,K,J)
9060  NEXT K
9070  NEXT J
9080  NEXT I
9085  REM PREMIERE SURFACE
9090  DATA  6
9100  DATA  0,0,0
9110  DATA  1,1,1
9120  DATA  1,0,2
9130  DATA  1,2,2
9140  DATA  1,2,0
9150  DATA  1,0,0
9155  REM SECONDE SURFACE
9160  DATA  6
9170  DATA  0,0,0
9180  DATA  1,0,2
9190  DATA  1,1,2
9200  DATA  1,2,1
9210  DATA  1,1,0
9220  DATA  1,0,0
9230  RETURN : REM ----------------------------------------------------
9240  REM
```



HACHURAGE DE SURFACES

Maintenant que nous sommes en mesure de positionner les divers constituants d'une scène, nous allons aborder la première étape du coloriage : **le hachurage de surfaces;**

HYPOTHESE

C'est au programme du chapitre précédent que nous allons ajouter cette nouvelle fonction. Ainsi, l'ensemble des sous-programmes que nous avons réalisé jusqu'à présent va nous servir à constituer une nouvelle scène dans laquelle les surfaces pourront être hachurées.

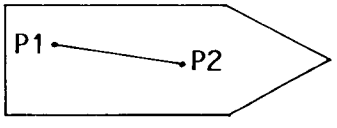
MODELE

Voyons en premier lieu, quelques notions qui nous serviront par la suite :

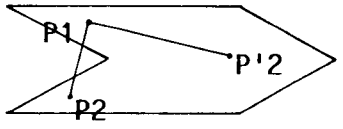
. Surface convexe :

Une surface S est convexe si pour tout couple (P1,P2) de point de S, le segment d'extrémités P1 et P2 est contenu dans S

Par exemple :



Surface convexe



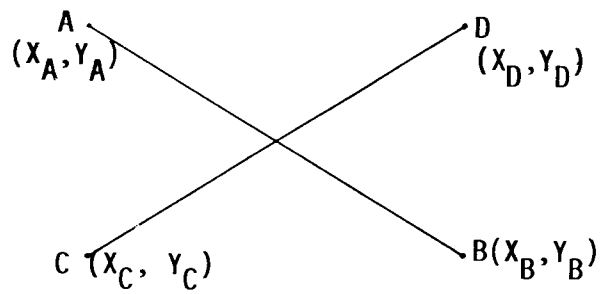
Surface non convexe

Cette définition est de toute première importance car l'algorithme de hachurage que nous allons développer dans ce chapitre ne fonctionnera que pour des surfaces de type convexe. A part quelques exceptions comme celles que l'on peut voir dans les exemples d'exécution des programmes, lorsque nous parlerons par la suite de surfaces, nous considérerons la propriété de convexité vérifiée.

. Intersection de deux segments

De nombreuses méthodes permettent le calcul des points d'intersection entre deux segments, mais nous avons choisi d'appliquer la méthode paramétrique décrite ci-dessous :

Considérons les deux segments AB et CD.



Les équations paramétriques de ces deux segments sont :

. **pour AB**
$$X = (Xb-Xa)T1 + Xa$$
$$Y = (Xb-Ya)T1 + Ya$$

. **pour CD**
$$X = (Xd-Xc)T2 + Xc$$
$$Y = (Yd-Yc)T2 + Yc$$

avec T1 et T2 compris entre les valeurs 0 et 1.

Pour trouver un éventuel point d'intersection entre les deux segments, on exprime le fait qu'il y a égalité des coordonnées pour les deux équations.

On a donc :

$$(Xb-Xa)\ T1+Xa = (Xd-Xc)\ T2+Xc$$
$$(Yb-Ya)\ T1+Ya = (Yd-Yc)\ T2+Yc$$

Ce qui s'écrit encore :

$$(Xb-Xa)\ T1+(Xc-Xd)\ T2 = Xc-Xa$$
$$(Yb-Ya)\ T1+(Yc-Yd)\ T2 = Yc-Ya$$

On cherche alors le "déterminant" du système d'équations à deux inconnues T1 et T2.

$$Det = \begin{bmatrix} Xb-Xa & Xc-Xd \\ Yb-Ya & Yc-Yd \end{bmatrix} = (Xb-Xa)\ (Yc-Yd) - (Xc-Xd)\ (Yb-Ya)$$

et si Det = Ø on conclut que les segments sont parallèles.
Sinon, on calcule :

$$T1 = \frac{1}{Det}\begin{bmatrix} Xc-Xa & Xc-Xd \\ Yc-Ya & Yc-Yd \end{bmatrix} = \frac{(Xc-Xa)\,(Y-Ya) - (Yc-Ya)\,(Xc-Xa)}{Det}$$

$$T2 = \frac{1}{Det}\begin{bmatrix} Xb-Xa & Xc-Xa \\ Yb-Ya & Yc-Ya \end{bmatrix} = \frac{(Xb-Xa)\,(Yc-Ya) - (Yb-Ya)\,(Xc-Xa)}{Det}$$

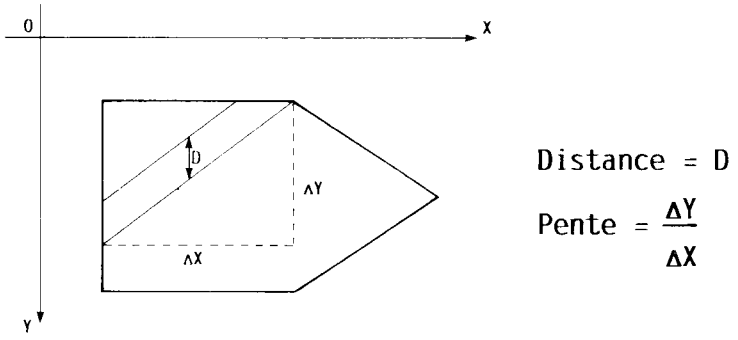
Alors si $\emptyset \leqslant T1 \leqslant 1$ le point cherché est sur le segment **AB**
et si $\emptyset \leqslant T2 \leqslant 1$ le point cherché est sur le segment **CD**

Et enfin, si les deux conditions sont vérifiées, c'est le point d'intersection qui est trouvé et qui a pour coordonnées :

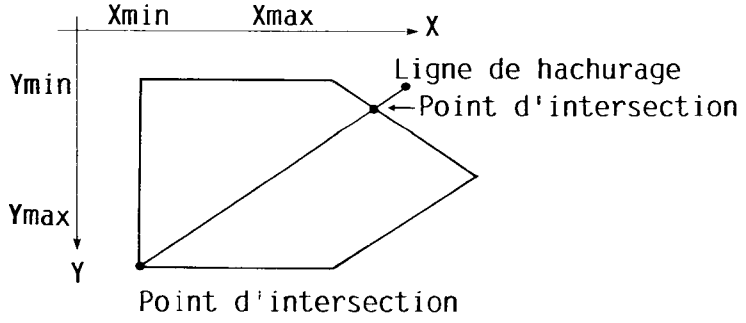
$$\left.\begin{aligned} X &= (Xb-Xa)\,T1 + Xa \\ Y &= (Yb-Ya)\,T1 + Ya \end{aligned}\right\} \text{obtenues en remplaçant T1 par la valeur calculée}$$

Cette méthode de calcul du point d'intersection nous sera fort utile pour le hachurage, car c'est entre deux points d'intersection que seront tracées les hachures.

Pour le hachurage proprement dit, chaque surface est considérée comme une entité indépendante. Chaque polygone peut donc être hachuré de façon autonome. Pour cela, nous définissons deux paramètres : une pente et une distance comme sur le schéma suivant :



Avant de commencer le hachurage d'une surface, la recherche de ses coordonnées minimum et maximum suivant les deux axes est effectuée. Cela est utile pour la future recherche des points d'intersection entre les lignes de hachurage de pente P et le polygone, contour de la surface, comme sur le schéma qui suit:



Xmin, **Xmax**, **Ymin** et **Ymax** sont déterminés par le sous-programme ***MIN-MAX.*** Les points d'intersection sont calculés dans le sous-programme ***POINTS D'INTERSECTION.***

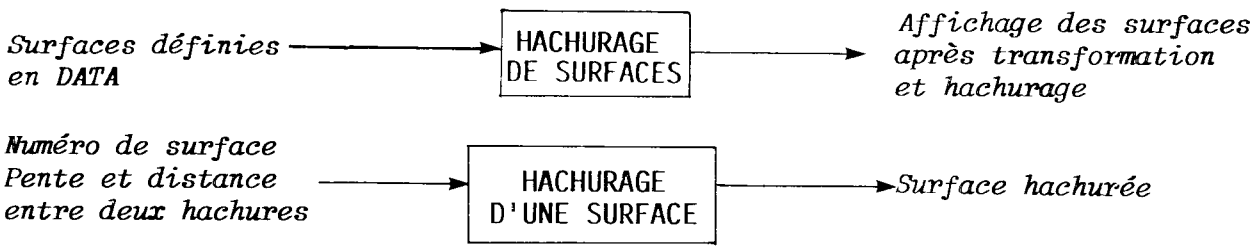
Lorsque les points d'intersection ont été déterminés, ils constituent une liste qui est triée par ordonnée croissante. Ceci est réalisé par le sous-programme ***TRI LISTE.*** Cette liste est ensuite testée afin d'éliminer les points doubles provenant des points de concours des segments et, afin de gérer certaines exceptiors. Après la réalisation de l'ensemble de ces opérations, une ligne de hachurage peut être tracée, entre deux points d'intersection, à l'aide du sous-programme ***VECTEUR.***

EXEMPLE D'EXECUTION

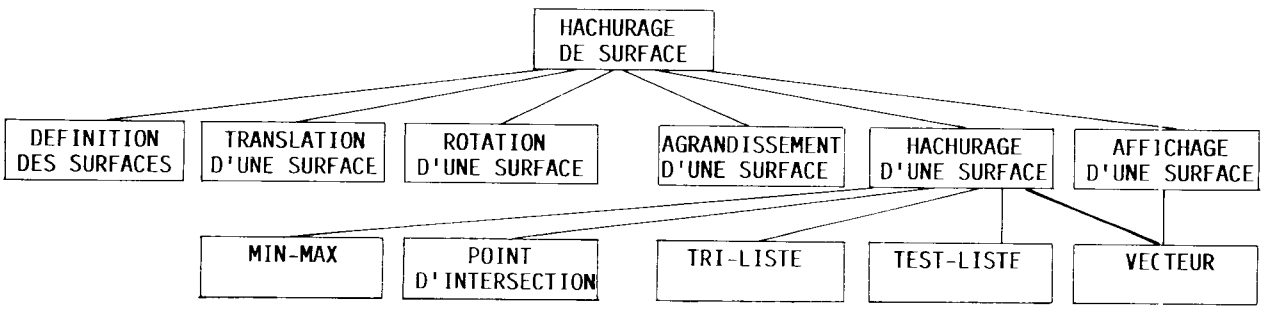
Nous allons voir en fait, deux exemples d'exécution. Le premier concerne le hachurage de deux surfaces convexes classiques. Le second exemple, quant à lui, met en oeuvre beaucoup plus avant certaines des transformations que nous avons vues jusqu'à présent. A noter, tout particulièrement, l'usage d'une même surface transformée en deux points différents de l'écran et les exceptions faites au niveau des surfaces qui ne sont pas toutes concernées par le hachurage.

LE PROGRAMME

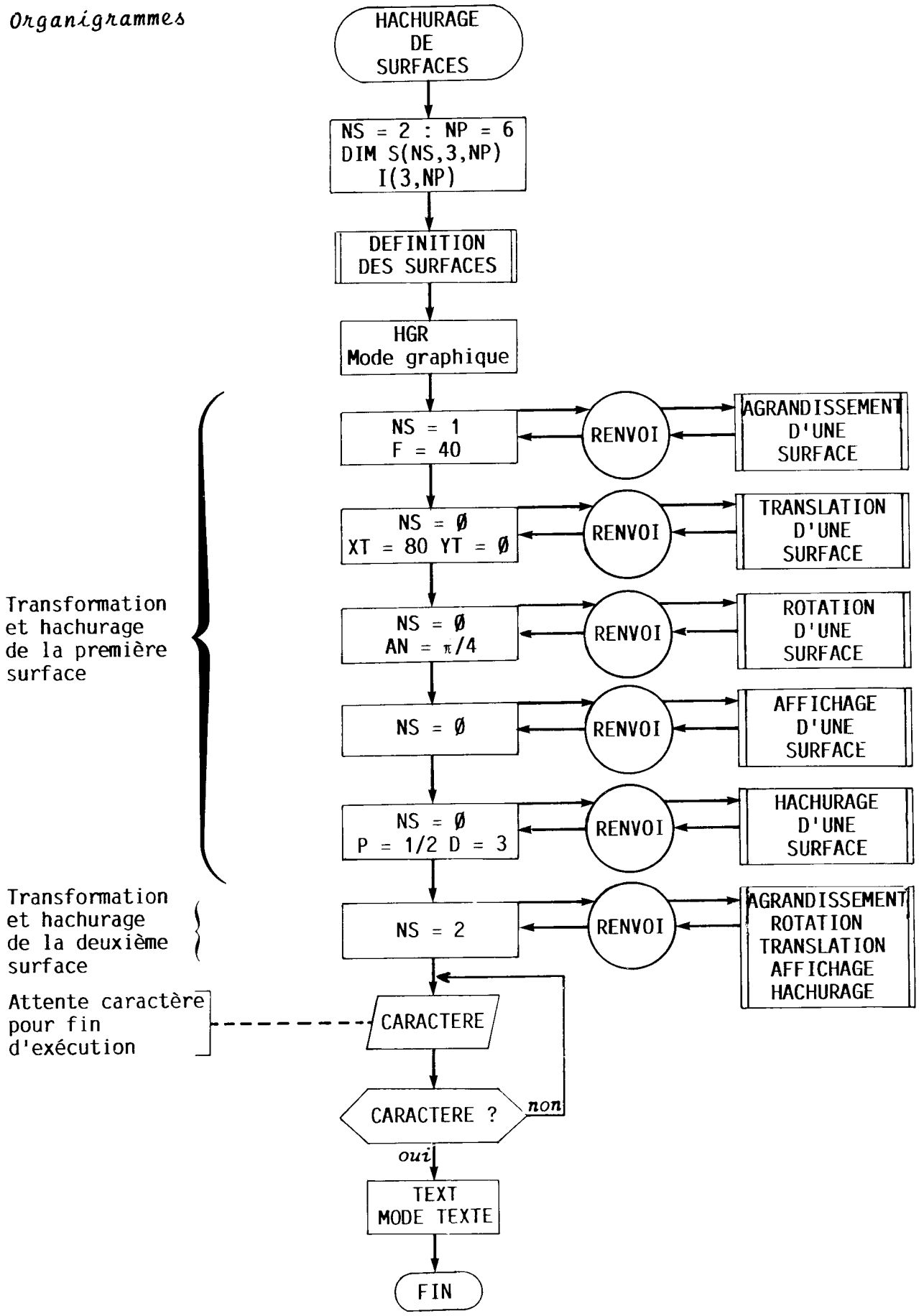
Traitement

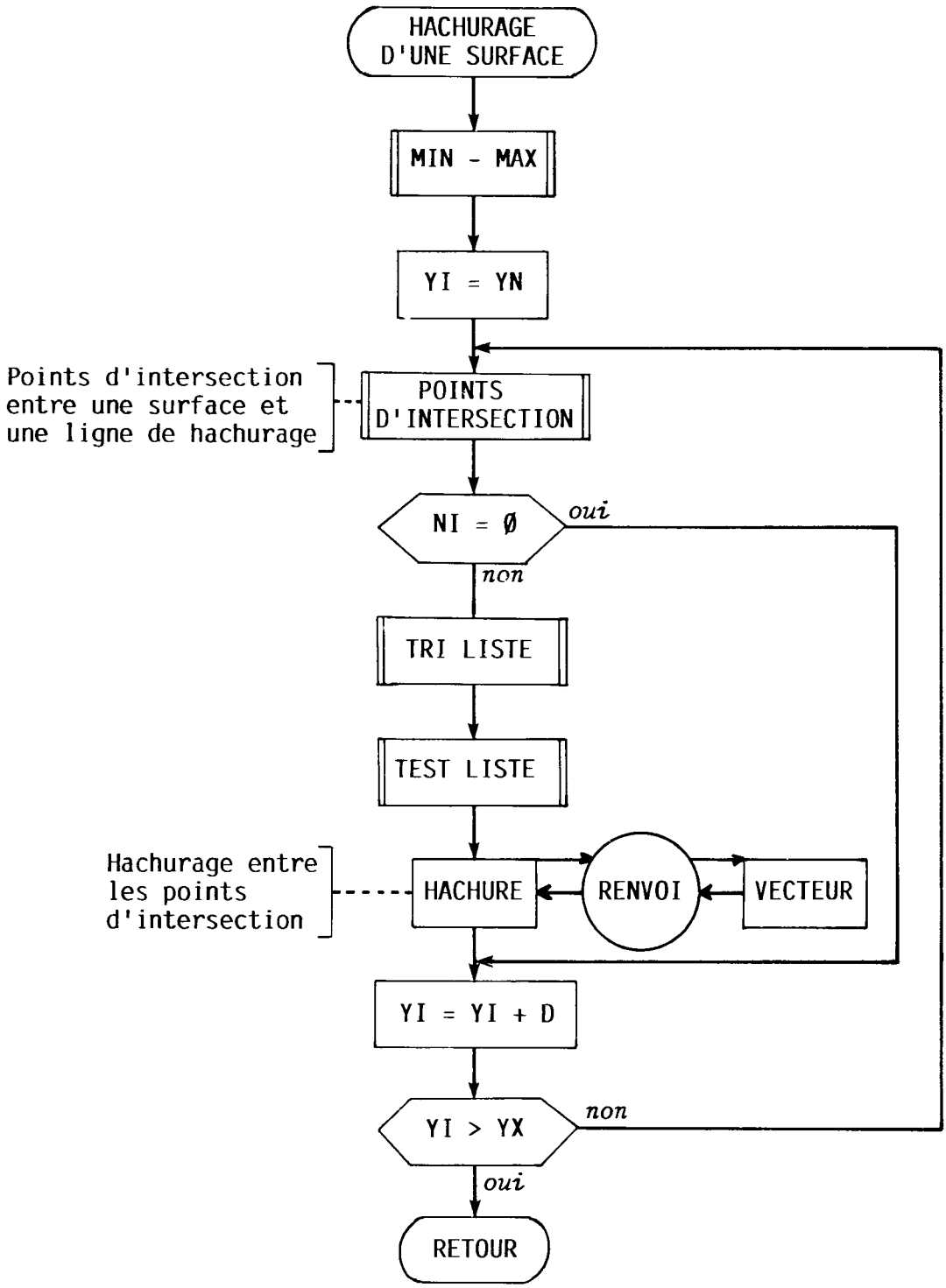


Structure



Organigrammes




HACHURAGE D'UNE SURFACE
MIN - MAX
YI = YN
Points d'intersection entre une surface et une ligne de hachurage
POINTS D'INTERSECTION
NI = Ø
oui
non
TRI LISTE
TEST LISTE
Hachurage entre les points d'intersection
HACHURE
RENVOI
VECTEUR
YI = YI + D
YI > YX
non
oui
RETOUR

Variables

HACHURAGE D'UNE SURFACE	:	Sous-programme principal du hachurage
YI	:	Ordonnée courante du hachurage
YN,YX	:	Limites du hachurage
NI	:	Nombre de points d'intersection calculés
I	:	Indice du tableau des points d'intersection
I (3,NP)	:	Tableau des points d'intersection
V1...V4	:	Coordonnées transférées au sous-programme ***VECTEUR***
MIN-MAX	:	Calcul des coordonnées limites de hachurage
XX,XN,YX,YN	:	Coordonnées limites
S (NS,3,NP)	:	Tableau des points d'une surface
NS	:	Numéro de surface
P	:	Pente des lignes de hachurage
I	:	Indice de boucle
POINTS D'INTERSECTION	:	Calcul des points d'intersection entre une ligne de hachurage et une surface.
NI	:	Nombre de points d'intersection calculés
NP	:	Nombre de points de la surface
S (NS,3,NP)	:	Tableau des points de la surface
F1...F6	:	Facteurs de calcul des points d'intersection
DE	:	Indicateurs de parallélisme
TX,TY	:	Indicateurs d'appartenance aux segments
P	:	Pente des lignes de hachurage
XX,XN	:	Abscisses limites des lignes de hachurage
YI	:	Ordonnée courante du hachurage
I (3,NP)	:	Tableau des points d'intersection
TRI LISTE	:	Tri des points d'intersection par ordonnée croissante
I	:	Indice du tableau des points
I (3,NP)	:	Tableau des points d'intersection à trier
Z	:	Variable de permutation
F	:	Indicateur de permutation
NI	:	Nombre de points d'intersection
TEST LISTE	:	Gestion des points doubles et des exceptions
I	:	Indice de boucle sur les points d'intersection
Z1,Z2	:	Coordonnées des points
I (3,NP)	:	Tableau des points d'intersection
J	:	Indice de boucle sur les points de la surface

K	:	Variable permettant de fermer un contour
F1...F6,DE	:	Calcul d'intersections visant à gérer les exceptions de convexité
TX,TY		
NP	:	Nombre de points du contour

AUTRES SOUS-PROGRAMMES : Voir ***AGRANDISSEMENT D'UNE SURFACE***

```
]LIST1000-

1000  REM ------------------------------------------------------------
1002  REM HACHURAGE DE SURFACES
1004  REM ------------------------------------------------------------
1010  REM NOMBRE DE SURFACES
1020 NS = 2
1025  REM NOMBRE MAXIMUM DE POINTS
1030 NP = 6
1035  REM DECLARATION DES TABLEAUX
1040  DIM S(NS,3,NP),I(3,NP)
1045  REM DEFINITION DES SURFACES
1050  GOSUB 11000: REM * DEFINITION DES SURFACES *
1055  REM PAGE 1 GRAPHIQUE
1060  HGR : POKE  - 16302,0: HCOLOR= 3
1065  REM AFFICHAGE DES SURFACES
1070 NS = 1:F = 40
1080  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1090 NS = 0:XT = 80:YT = 0
1100  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1110 NS = 0:AN = 3.14159 / 4
1120  GOSUB 6000: REM * ROTATION D'UNE SURFACE *
1130 NS = 0: GOSUB 8000: REM * AFFICHAGE D'UNE SURFACE *

1140 NS = 0:P = 1 / 2:D = 3
1150  GOSUB 9000: REM * HACHURAGE D'UNE SURFACE *
1160 NS = 2:F = 60
1170  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1180 NS = 0:AN = 3.14159 / 2
1190  GOSUB 6000: REM * ROTATION D'UNE SURFACE *
1200 NS = 0:XT = 255:YT = 60
1210  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1220 NS = 0: GOSUB 8000: REM * AFFICHAGE D'UNE SURFACE *

1230 NS = 0:P = 1 / 4:D = 3
1240  GOSUB 9000: REM * HACHURAGE D'UNE SURFACE *
1250  GET R$: IF R$ = "" THEN 1250
1260  HOME : TEXT
1270  END : REM ------------------------------------------------------
1280  REM
1290  REM
4000  REM ------------------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ----------------------------------------------------
4050  REM
4060  REM
```

```
5000  REM ------------------------------------------------------------
5002  REM TRANSLATION D'UNE SURFACE
5004  REM ------------------------------------------------------------
5010  FOR I = 1 TO NP
5020 S(0,1,I) = S(NS,1,I)
5030 S(0,2,I) = S(NS,2,I) + XT
5040 S(0,3,I) = S(NS,3,I) + YT
5050  NEXT I
5060  RETURN : REM ------------------------------------------------------
5070  REM
5080  REM
6000  REM ------------------------------------------------------------
6002  REM ROTATION D'UNE SURFACE
6004  REM ------------------------------------------------------------
6010  FOR I = 1 TO NP
6020 XP = S(NS,2,I) *  COS (AN) - S(NS,3,I) *  SIN (AN)
6030 YP = S(NS,2,I) *  SIN (AN) + S(NS,3,I) *  COS (AN)
6040 S(0,2,I) = XP:S(0,3,I) = YP
6050 S(0,1,I) = S(NS,1,I)
6060  NEXT I
6070  RETURN : REM ------------------------------------------------------
6080  REM
6090  REM
7000  REM ------------------------------------------------------------
7002  REM AGRANDISSEMENT D'UNE SURFACE
7004  REM ------------------------------------------------------------
7010  FOR I = 1 TO NP
7020 S(0,1,I) = S(NS,1,I)
7030 S(0,2,I) = S(NS,2,I) * F
7040 S(0,3,I) = S(NS,3,I) * F
7050  NEXT I
7060  RETURN : REM ------------------------------------------------------
7070  REM
7080  REM
8000  REM ------------------------------------------------------------
8002  REM AFFICHAGE D'UNE SURFACE
8004  REM ------------------------------------------------------------
8010  FOR I = 1 TO NP
8020  IF S(NS,1,I) = 1 THEN 8050
8030 V3 = S(NS,2,I):V4 = S(NS,3,I)
8040  GOTO 8080
8050 V1 = V3:V2 = V4
8060 V3 = S(NS,2,I):V4 = S(NS,3,I)
8070  GOSUB 4000: REM * VECTEUR *
8080  NEXT I
8090  RETURN : REM ------------------------------------------------------
8100  REM
8110  REM
```

```
9000  REM ------------------------------------------------------------
9002  REM HACHURAGE D'UNE SURFACE
9004  REM ------------------------------------------------------------
9010  GOSUB 9300: REM * MIN-MAX *
9020  FOR YI = YN TO YX STEP D
9030  GOSUB 9500: REM * INTERSECTIONS *
9040  IF NI = 0 THEN 9160
9050  GOSUB 9700: REM * TRI LISTE  *
9060  GOSUB 9900: REM * TEST LISTE *
9070 I = 0
9080 I = I + 1: IF I > NI THEN 9160
9090  IF I(1,I) = 0 THEN 9150
9100 V1 = I(2,I):V2 = I(3,I)
9110 I = I + 1: IF I > NI THEN 9160
9120  IF I(1,I) = 0 THEN 9110
9130 V3 = I(2,I):V4 = I(3,I)
9140  GOSUB 4000: REM * VECTEUR *
9150  IF I < NI THEN 9090
9160  NEXT YI
9170  RETURN : REM ------------------------------------------------------
9180  REM
9190  REM
9300  REM ------------------------------------------------------------
9302  REM MIN-MAX
9304  REM ------------------------------------------------------------
9310 XX = S(NS,2,1):XN = S(NS,2,1)
9320 YX = S(NS,3,1):YN = S(NS,3,1)
9330  FOR I = 2 TO NP
9340  IF S(NS,2,I) > XX THEN XX = S(NS,2,I): GOTO 9360
9350  IF S(NS,2,I) < XN THEN XN = S(NS,2,I)
9360  IF S(NS,3,I) > YX THEN YX = S(NS,3,I): GOTO 9380
9370  IF S(NS,3,I) < YN THEN YN = S(NS,3,I)
9380 YI = S(NS,3,I) - P * S(NS,2,I)
9390  IF YI > YX THEN YX = YI: GOTO 9410
9400  IF YI < YN THEN YN = YI
9410  NEXT I
9420  REM ------------------------------------------------------------
9430  REM
9440  REM
9500  REM ------------------------------------------------------------
9502  REM POINTS D'INTERSECTION
9504  REM ------------------------------------------------------------
9510 NI = 0
9520  FOR I = 1 TO NP - 1
9530 F1 = S(NS,2,I) - S(NS,2,I + 1):F2 = S(NS,3,I) - S(N
    S,3,I + 1)
9540 F3 = S(NS,2,I) - XN:F4 = S(NS,3,I) - YI
```

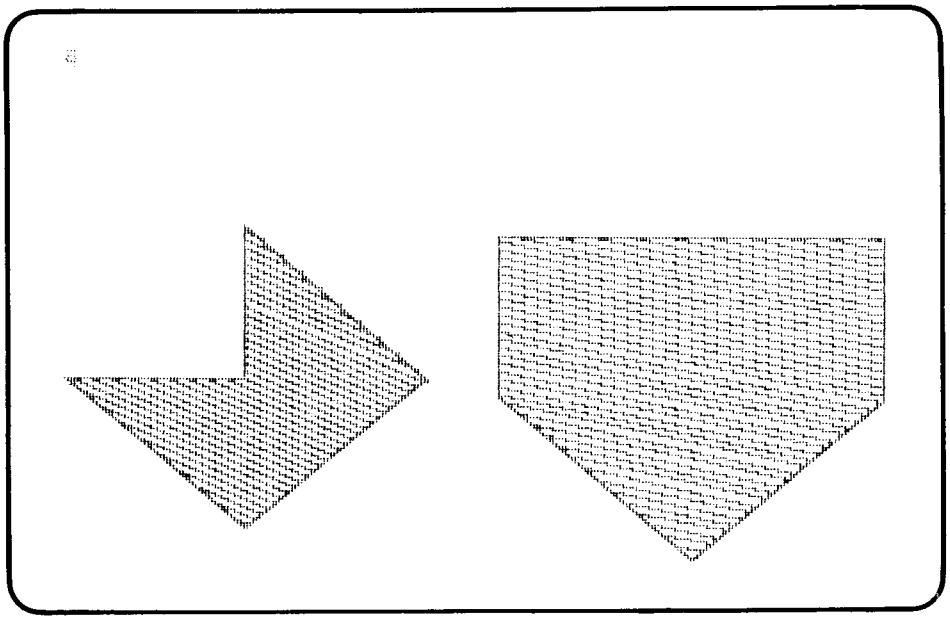
```
9550 F5 = XX - XN:F6 = F5 * P
9560 DE = F5 * F2 - F1 * F6
9570  IF DE = 0 THEN 9650
9580 TX = (F3 * F2 - F1 * F4) / DE
9590 TY = (F5 * F4 - F3 * F6) / DE
9600  IF TX < 0 OR TX > 1 THEN 9650
9610  IF TY < 0 OR TY > 1 THEN 9650
9620 NI = NI + 1
9630 I(2,NI) = F5 * TX + XN
9640 I(3,NI) = F6 * TX + YI
9650  NEXT I
9660  RETURN : REM ----------------------------------------
9670  REM
9680  REM
9700  REM ----------------------------------------
9702  REM TRI LISTE
9704  REM ----------------------------------------
9710 I = 1
9720  IF I(3,I) <  = I(3,I + 1) THEN 9760
9730 Z = I(2,I):I(2,I) = I(2,I + 1):I(2,I + 1) = Z
9740 Z = I(3,I):I(3,I) = I(3,I + 1):I(3,I + 1) = Z
9750 F = 1
9760  IF I < NI - 1 THEN I = I + 1: GOTO 9720
9770  IF F <  > 0 THEN F = 0: GOTO 9710
9780  RETURN : REM ----------------------------------------
9790  REM
9800  REM
9900  REM ----------------------------------------
9902  REM TEST LISTE
9904  REM ----------------------------------------
9910  FOR I = 1 TO NI
9920 Z1 =  INT (I(3,I) + .5):Z2 =  INT (I(3,I + 1) + .5)

9930  IF Z1 <  > Z2 THEN I(1,I) = 1: GOTO 10140
9940 Z1 =  INT (I(2,I) + .5):Z2 =  INT (I(2,I + 1) + .5)

9950  IF Z1 <  > Z2 THEN I(1,I) = 1: GOTO 10140
9960  FOR J = 1 TO NP
9970  IF  INT (S(NS,3,J) + .5) <  >  INT (I(3,I) + .5) THEN
    10130
9980  IF  INT (S(NS,2,J) + .5) <  >  INT (I(2,I) + .5) THEN
    10130
9990 K = J: IF J = 1 THEN K = NP + 1
10000 F1 = S(NS,2,K - 1) - S(NS,2,J + 1)
10010 F2 = S(NS,3,K - 1) - S(NS,3,J + 1)
10020 F3 = S(NS,2,K - 1) - XN
```

```
10030 F4 = S(NS,3,K - 1) - Y1
10040 DE = F5 * F2 - F1 * F6
10050  IF DE = 0 THEN 10130
10060 TX = (F3 * F2 - F1 * F4) / DE
10070 TY = (F5 * F4 - F3 * F6) / DE
10080  IF TX < 0 OR TX > 1 THEN 10120
10090  IF TY < 0 OR TY > 1 THEN 10120
10100 I(1,I) = 1:I(1,I + 1) = 0:I = I + 1:J = NP
10110  GOTO 10130
10120 I(1,I) = 0:I(1,I + 1) = 0:I = I + 1:J = NP
10130  NEXT J
10140  NEXT I
10150  RETURN : REM ------------------------------------------------
10160  REM
10170  REM
11000  REM ------------------------------------------------
11002  REM DEFINITION DES SURFACES
11004  REM ------------------------------------------------
11010  FOR I = 1 TO NS
11020  READ NP
11030  FOR J = 1 TO NP
11040  FOR K = 1 TO 3
11050  READ S(I,K,J)
11060  NEXT K
11070  NEXT J
11080  NEXT I
11085  REM PREMIERE SURFACE
11090  DATA  6
11100  DATA  0,0,0
11110  DATA  1,1,1
11120  DATA  1,0,2
11130  DATA  1,2,2
11140  DATA  1,2,0
11150  DATA  1,0,0
11155  REM DEUXIEME SURFACE
11160  DATA  6
11170  DATA  0,0,0
11180  DATA  1,0,2
11190  DATA  1,1,2
11200  DATA  1,2,1
11210  DATA  1,1,0
11220  DATA  1,0,0
11230  RETURN : REM ------------------------------------------------
11240  REM
```

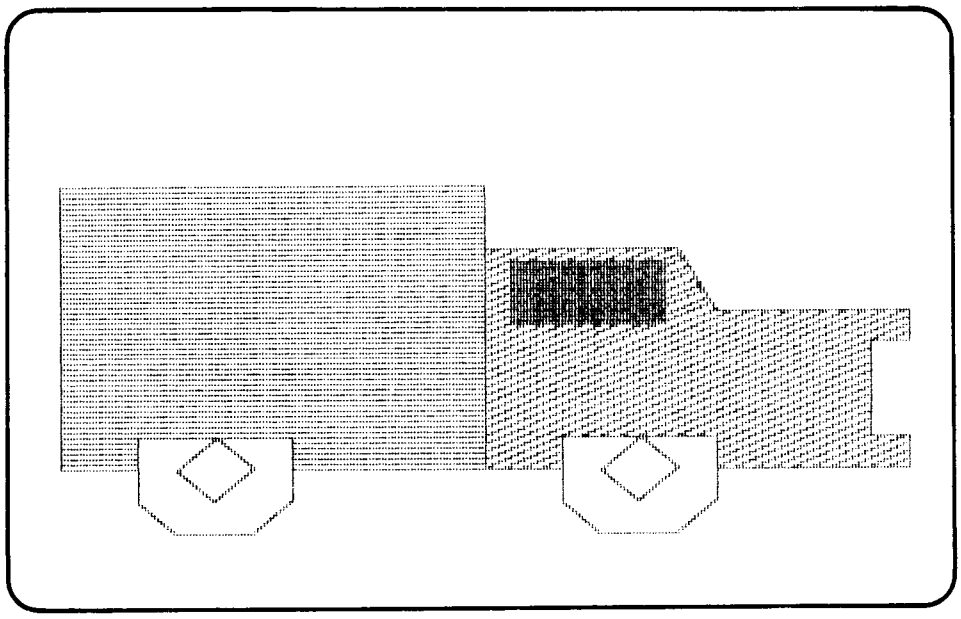
```
]LIST1000-1400

1000  REM ------------------------------------------------------------
1002  REM EXEMPLE DE HACHURAGE
1004  REM ------------------------------------------------------------
1010  REM NOMBRE DE SURFACES
1020 NS = 4
1025  REM NOMBRE MAXIMUM DE POINTS
1030 NP = 15
1035  REM DECLARATION DES TABLEAUX
1040  DIM S(NS,3,NP),I(3,NP)
1045  REM DEFINITION DES SURFACES
1050  GOSUB 11000: REM * DEFINITION DES SURFACES *
1055  REM PAGE 1 GRAPHIQUE
1060  HGR : POKE  - 16302,0: HCOLOR= 3
1065  REM AFFICHAGE DES SURFACES
1070 NS = 1:NP = 9:F = 12
1080  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1090 NS = 0:XT = 0:YT = 36
1100  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1110 NS = 0:NP = 9: GOSUB 8000: REM * AFFICHAGE D'UNE SU
     RFACE *
1120 NS = 0:P = 0:D = 2
1130  GOSUB 9000: REM * HACHURAGE D'UNE SURFACE *
1140 NS = 2:NP = 15:F = 12
1150  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1160 NS = 0:XT = 132:YT = 60
1170  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1180 NS = 0: GOSUB 8000: REM * AFFICHAGE D'UNE SURFACE *

1190 NS = 0:P =  - 1 / 2:D = 3
1200  GOSUB 9000: REM * HACHURAGE D'UNE SURFACE *
1210 NS = 3:NP = 12:F = 12
1220  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1230 NS = 0:XT = 24:YT = 132
1240  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1250 NS = 0: GOSUB 8000: REM * AFFICHAGE D'UNE SURFACE *

1260 NS = 0:XT = 132:YT = 0
1270  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1280 NS = 0: GOSUB 8000: REM * AFFICHAGE D'UNE SURFACE *

1290 NS = 4:NP = 5:F = 12
1300  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1310 NS = 0:XT = 140:YT = 65
1320  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1330 NS = 0: GOSUB 8000: REM * AFFICHAGE D'UNE SURFACE *

1340 NS = 0:P = 0:D = 1
1350  GOSUB 9000: REM * HACHURAGE D'UNE SURFACE *
1360  GET R$: IF R$ = "" THEN 1360
1370  HOME : TEXT
1380  END : REM ------------------------------------------------------
1390  REM
1400  REM
```

```
11000  REM ------------------------------------------------
11002  REM DEFINITION DES SURFACES
11004  REM ------------------------------------------------
11010  FOR I = 1 TO N5
11020  READ NP
11030  FOR J = 1 TO NP
11040  FOR K = 1 TO 3
11050  READ S(I,K,J)
11060  NEXT K
11070  NEXT J
11080  NEXT I
11085  REM PREMIERE SURFACE
11090  DATA  9
11100  DATA  0,0,0
11110  DATA  1,11,0
11120  DATA  1,11,9
11130  DATA  1,6,5
11140  DATA  1,6,8
11150  DATA  1,2,8
11160  DATA  1,2,9
11170  DATA  1,0,9
11180  DATA  1,0,0
11185  REM DEUXIEME SURFACE
11190  DATA  15
11200  DATA  0,0,0
11210  DATA  1,5,0
11220  DATA  1,6,2
11230  DATA  1,11,2
11240  DATA  1,11,3
11250  DATA  1,10,3
11260  DATA  1,10,6
11270  DATA  1,11,6
11280  DATA  1,11,7
11290  DATA  1,6,7
11300  DATA  1,6,6
11310  DATA  1,2,6
11320  DATA  1,2,5
11330  DATA  1,0,5
11340  DATA  1,0,0
11345  REM TROISIEME SURFACE
11350  DATA  12
11360  DATA  0,0,0
11370  DATA  1,4,0
11380  DATA  1,4,2
11390  DATA  1,3,3
11400  DATA  1,1,3
11410  DATA  1,0,2
11420  DATA  1,0,0
11430  DATA  0,2,0
11440  DATA  1,3,1
11450  DATA  1,2,2
11460  DATA  1,1,1
11470  DATA  1,2,0
11475  REM QUATRIEME SURFACE
11480  DATA  5
11490  DATA  0,0,0
11500  DATA  1,4,0
11510  DATA  1,4,2
11520  DATA  1,0,2
11530  DATA  1,0,0
11540  RETURN : REM ----------------------------------------
11550  REM
```

Exemple d'hachurage



Hachurage d'une surface



ÉLIMINATION DES PARTIES CACHÉES

Cette fois, nous allons améliorer les conditions de visualisation des scènes en travaillant en deux dimensions et demie. C'est ainsi que les spécialistes définissent l'espace de travail qui tient compte des priorités d'affichage sur l'écran et conduit à l'élimination des parties cachées.

HYPOTHESE

Pour mettre en oeuvre l'algorithme d'élimination des parties cachées, nous allons reprendre le programme écrit pour l'exemple d'exécution : ***AGRANDISSEMENT DF SURFACE.*** Nous définirons des surfaces se chevauchant en utilisant le numéro de celles-ci en tant qu'indice de priorité. Ainsi, la première surface sera toujours entièrement visible, alors que les suivantes pourront être cachées partiellement selon leur disposition dans la scène.

MODELE

Lorsque nous ne voulions pas éliminer les parties cachées des surfaces, chaque contour était traité de façon indépendante. Maintenant, il existe entre chaque surface une relation de priorité et nous sommes contraints de les tracer en tenant compte de chacune d'elles. Ceci nous a amené à mettre en forme le sous-programme ***COPIE DE SURFACE*** qui permet, après avoir transformé un contour, de le stocker dans le tableau des points de surface.

Dans le programme, le stockage est réalisé sous le même indice que celui qui est utilisé pour la lecture, cependant, on aurait pu imaginer de surdimensionner le tableau afin de stocker les

contours sous des numéros de surface différents. En outre, nous aurions pu aussi, stocker le nombre de points de chaque surface ainsi que la priorité de la surface dans le tableau S (NTS,3,NP). Ces diverses fonctionnalités ne sont pas réalisées pour des raisons de diminution de l'espace mémoire utile et parce que le nombre de surfaces mises en jeu est faible. Néanmoins, elles peuvent dans certains cas, présenter un avantage réel.

Ces mises au point étant faites, revenons à l'élimination des parties cachées. Après avoir transformé chaque surface, nous disposons dans le tableau S (NS,3,NP) d'une image virtuelle de ce qui serait affiché sur l'écran sans élimination. Alors, pour éliminer les éléments cachés, nous allons calculer les intersections entre une ligne parallèle à l'axe-X et les surfaces à afficher. Cette ligne débute sa course en haut de la partie visible des dessins et parcourt l'axe-Y d'unité en unité jusqu'à la fin de cette même partie visible. Les limites du dessin sont déterminées par le sous-programme ***ORDONNEES MIN ET MAX.***

A chaque hauteur de sa course, la ligne d'observation coupe les surfaces en certains points. Ces points d'intersection, déterminés par le sous-programme ***INTERSECTIONS*** sont stockés dans une liste qui est triée par ordre d'abscisse croissante par le sous-programme ***TRI LISTE.*** La liste triée est ensuite fournie au sous-programme ***AFFICHER*** qui réalise l'essentiel de l'élimination des parties cachées grâce au sous-programme ***PUSH*** qui n'aurait pas été nécessaire avec un langage récursif.

Voyons sur un exemple comment le sous-programme ***AFFICHER***, procède à partir de la liste des points d'intersection suivante :

Points (X,Y)	P1	P2	P3	P4	P5	P6
Priorité (NS)	3	1	1	2	3	2
Segment 1 = Vrai Ø = Faux	1	0	0	0	1	0

où les abscisses de P1 à P6 sont différentes

Processus d'élimination :

Point courant de AFFICHER	Point suivant (Liste)	Pile triée (PUSH)	Tracé	Raison NS = numéro de surface S = segment
P1	P2		P1•—→P2	NS(P1)>NS(P2) et S(P1)=1
P2	P3	P1	P1•—→P2• •P3	NS(P1)=NS(P3)
P1	P4		P1•—→P2• •P3—→P4	NS(P1)>NS(P4) et S(P1)=1
P4	P5	P1	P1•—→P2• •P3•—→P4	NS(P4)<NS(P5) et NS(P5)=NS(P1)
P4	P6		P1•—→P2• •P3—→P4• •P6	NS(P4)=NS(P6)

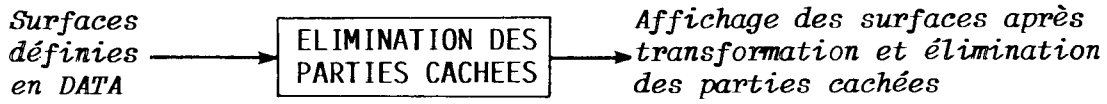
En répétant ce processus pour chaque ligne d'observation, l'élimination des parties cachées d'une scène est entièrement réalisée.

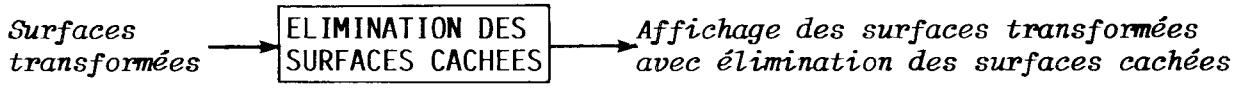
EXEMPLE D'EXECUTION

L'exemple d'exécution dont on peut voir le résultat en page 195 réalise l'élimination des parties cachées entre quatre surfaces qui se superposent partiellement.

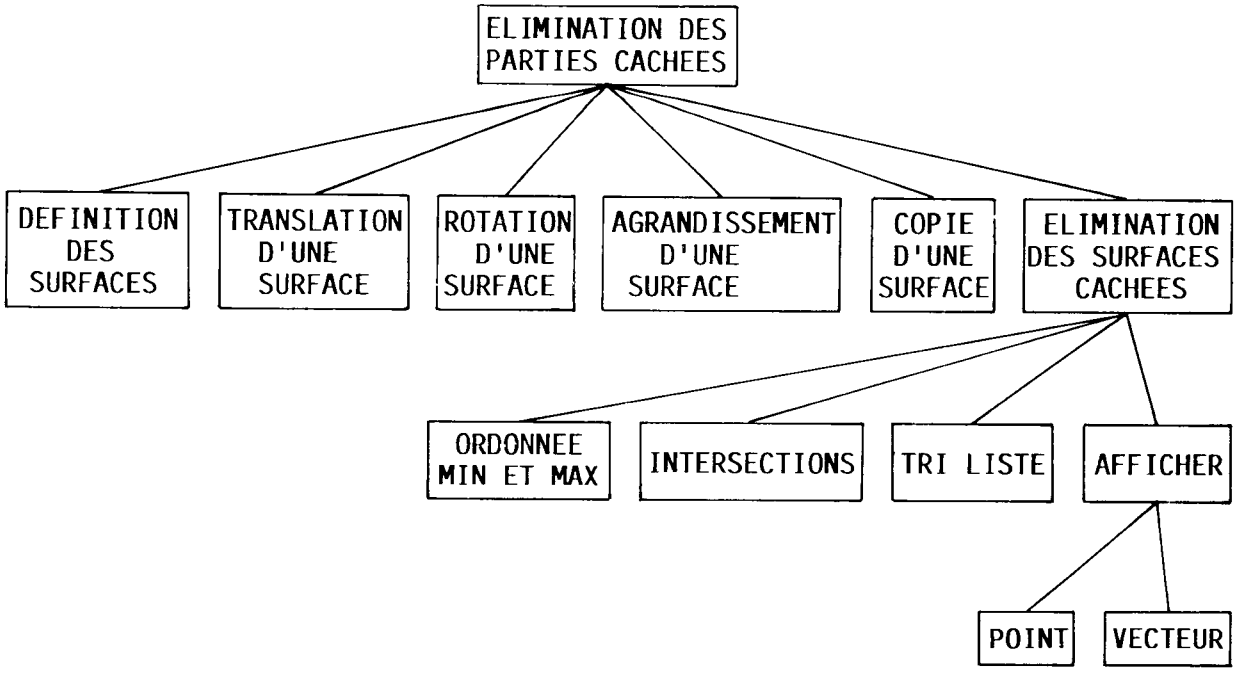
LE PROGRAMME

Traitement

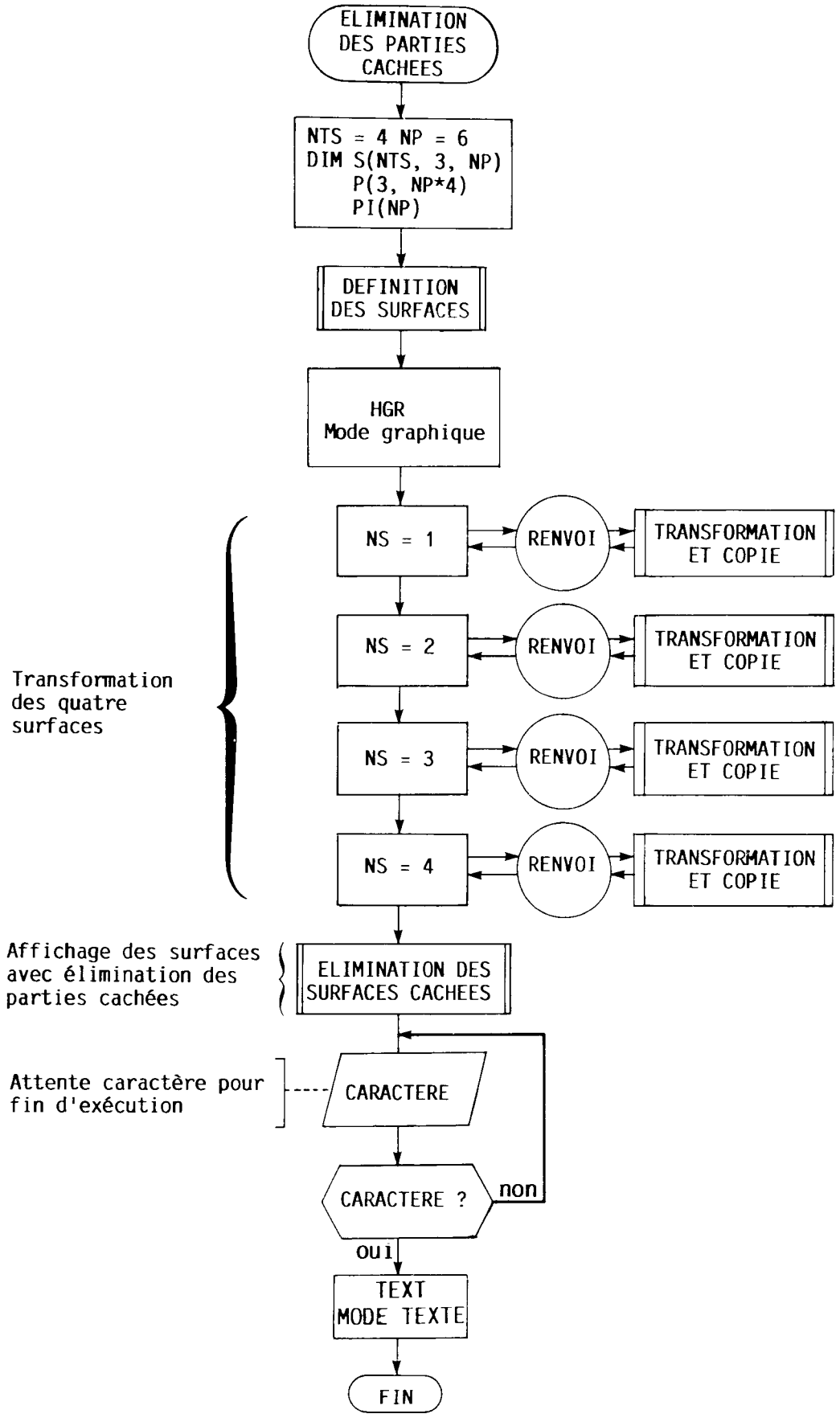


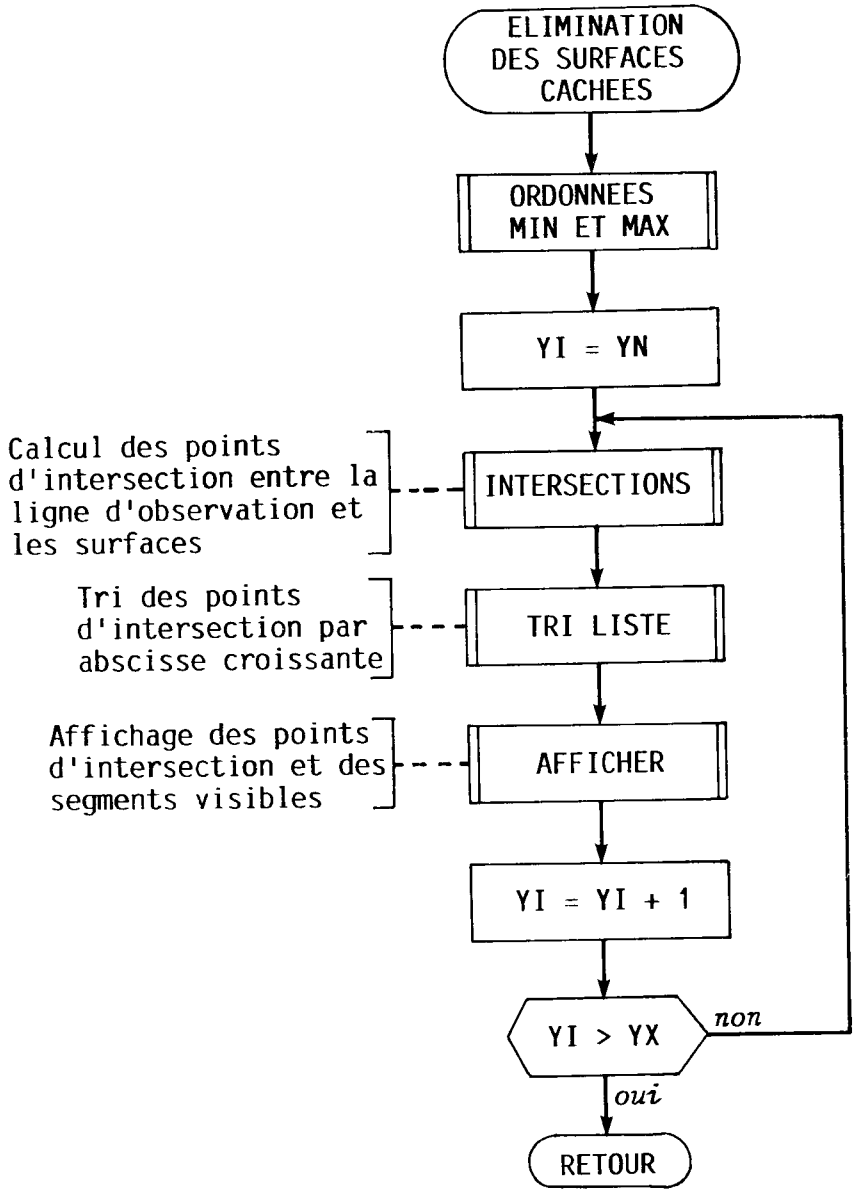


Structure



Organigrammes




ELIMINATION DES SURFACES CACHEES
ORDONNEES MIN ET MAX
YI = YN
Calcul des points d'intersection entre la ligne d'observation et les surfaces
INTERSECTIONS
Tri des points d'intersection par abscisse croissante
TRI LISTE
Affichage des points d'intersection et des segments visibles
AFFICHER
YI = YI + 1
YI > YX
non
oui
RETOUR

Variables

COPIE D'UNE SURFACE	:	Transfert d'une surface à l'intérieur du tableau S (NS,3,NP)
I	:	Indice assurant le transfert de tous les points
S (NS,3,NP)	:	Tableau des points des surfaces à transférer
ELIMINATION DES SURFACES CACHEES	:	Elimination à partir des surfaces transformées
YI	:	Ordonnée de la ligne d'observation
YN,YX	:	Ordonnées limites de la ligne d'observation
ORDONNEES MIN ET MAX	:	Recherche des ordonnées limites
YN	:	Ordonnée minimum
YX	:	Ordonnée maximum
I,J	:	Indices permettant le parcours de tous les contours de toutes les surfaces
S (NS,3,I)	:	Tableau des contours transformés
INTERSECTIONS	:	Calcule les intersections entre la ligne d'observation et les différentes surfaces
K	:	Nombre de points d'intersection calculés
I,J	:	Indices permettant le parcours de tous les contours de toutes les surfaces
S (NS,3,I)	:	Tableau des contours transformés
YI	:	Ordonnée courante de la ligne d'observation
F1...F5	:	Facteurs de calcul des points d'intersection
DE,TX,TY	:	Variables de calcul d'intersections
P (3,NP*4)	:	Tableau des points d'intersection calculés
TRI LISTE	:	Tri de la liste des points d'intersection par abscisse croissante
I,J	:	Indices du tableau des points d'intersection
P (3,NP*4)	:	Tableau des points d'intersection
Z	:	Variable de permutation
K	:	Nombre de points d'intersection restants après élimination des points double

AFFICHER	:	Trace les éléments visibles des contours
I,J	:	Indices de parcours des points d'intersection
L	:	Nombre de points stockés en pile
P (3,NP*4)	:	Tableau des points d'intersection
XI,YI	:	Abscisses et ordonnées courantes
PI (NP)	:	Tableau contenant l'indice du point d'intersection supposé être stocké en pile
M	:	Indice à stocker dans PI (NP)
V1...V4	:	Coordonnées transférées à VECTEUR
X,Y	:	Coordonnées transférées à POINT
PUSH	:	Met en pile les indices des points d'intersection en attente
II,JJ	:	Indices de parcours de la pile pour le tri et l'élimination des points
PI (NP)	:	Pile des indices des points d'intersection
L	:	Nombre d'éléments en pile
F	:	Indicateur de permutation
AUTRE SOUS-PROGRAMMES	:	Voir *AGRANDISSEMENT DE SURFACES*

```
§LIST1000-

1000  REM ------------------------------------------------
1002  REM ELIMINATION DES PARTIES CACHEES
1004  REM ------------------------------------------------
1010  REM NOMBRE TOTAL DE SURFACES
1020 NTS = 4
1025  REM NOMBRE MAXIMUM DE POINTS
1030 NP = 6
1035  REM DECLARATION DES TABLEAUX
1040  DIM S(NTS,3,NP),P(3,NP * 4),PI(NP)
1045  REM DEFINITION DES SURFACES
1050  GOSUB 12000: REM * DEFINITION DES SURFACES *
1055  REM PAGE 1 GRAPHIQUE
1060  HGR : POKE  - 16302,0: HCOLOR= 3
1065  REM TRANSFORMATION DES SURFACES
1066  REM PREMIERE SURFACE
1070 NS = 1:NP = 6:F = 40
1080  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1090 NS = 0:XT = 80:YT =  - 40
1100  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1110 NS = 0:AN = 3.14159 / 4
1120  GOSUB 6000: REM * ROTATION D'UNE SURFACE *
1130 NS = 1:NP = 6
1140  GOSUB 9000: REM * COPIE D'UNE SURFACE *
1145  REM DEUXIEME SURFACE
1150 NS = 2:NP = 6:F = 40
1160  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1170 NS = 0:AN = 3.14159 / 2
1180  GOSUB 6000: REM * ROTATION D'UNE SURFACE *
1190 NS = 0:XT = 240:YT = 80
1200  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1210 NS = 2:NP = 6
1220  GOSUB 9000: REM * COPIE D'UNE SURFACE *
1225  REM TROISIEME SURFACE
1230 NS = 3:NP = 5:F = 50
1240  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1250 NS = 0:XT = 60:YT = 38
1260  GOSUB 5000: REM *TRANSLATION D'UNE SURFACE *
1270 NS = 3:NP = 5
1280  GOSUB 9000: REM *COPIE D'UNE SURFACE *
1285  REM QUATRIEME SURFACE
1290 NS = 4:NP = 6:F = 72
1300  GOSUB 7000: REM * AGRANDISSEMENT D'UNE SURFACE *
1310 NS = 0:XT = 35:YT = 31
1320  GOSUB 5000: REM * TRANSLATION D'UNE SURFACE *
1330 NS = 4:NP = 6
1340  GOSUB 9000: REM * COPIE D'UNE SURFACE *
1345  REM AFFICHAGE AVEC PARTIES CACHEES
1350  GOSUB 10000: REM * ELIMINATION DES SURFACES CACHEE
    S *
1360  GET R$: IF R$ = "" THEN 1360
1370  HOME
1380  TEXT
1390  END : REM ------------------------------------------
1400  REM
1410  REM
```

```
3000  REM ------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ----------------------------------------
3070  REM
3080  REM
4000  REM ------------------------------------------------
4002  REM VECTEUR
4004  REM ------------------------------------------------
4005  REM V1,V2: COORDONNEES DU POINT 1
4006  REM V3,V4: COORDONNEES DU POINT 2
4010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
4020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
4030  HPLOT V1,V2 TO V3,V4
4040  RETURN : REM ----------------------------------------
4050  REM
4060  REM
5000  REM ------------------------------------------------
5002  REM TRANSLATION D'UNE SURFACE
5004  REM ------------------------------------------------
5010  FOR I = 1 TO NP
5020 S(0,1,I) = S(NS,1,I)
5030 S(0,2,I) = S(NS,2,I) + XT
5040 S(0,3,I) = S(NS,3,I) + YT
5050  NEXT I
5060  RETURN : REM ----------------------------------------
5070  REM
5080  REM
```

```
6000  REM ------------------------------------------------------
6002  REM ROTATION D'UNE SURFACE
6004  REM ------------------------------------------------------
6010  FOR I = 1 TO NP
6020 XP = S(NS,2,I) *  COS (AN) - S(NS,3,I) *  SIN (AN)
6030 YP = S(NS,2,I) *  SIN (AN) + S(NS,3,I) *  COS (AN)
6040 S(0,2,I) = XP:S(0,3,I) = YP
6050 S(0,1,I) = S(NS,1,I)
6060  NEXT I
6070  RETURN : REM ----------------------------------------------
6080  REM
6090  REM
7000  REM ------------------------------------------------------
7002  REM AGRANDISSEMENT D'UNE SURFACE
7004  REM ------------------------------------------------------
7010  FOR I = 1 TO NP
7020 S(0,1,I) = S(NS,1,I)
7030 S(0,2,I) = S(NS,2,I) * F
7040 S(0,3,I) = S(NS,3,I) * F
7050  NEXT I
7060  RETURN : REM ----------------------------------------------
7070  REM
7080  REM
8000  REM ------------------------------------------------------
8002  REM AFFICHAGE D'UNE SURFACE
8004  REM ------------------------------------------------------
8010  FOR I = 1 TO NP
8020  IF S(NS,1,I) = 1 THEN 8050
8030 V3 = S(NS,2,I):V4 = S(NS,3,I)
8040  GOTO 8080
8050 V1 = V3:V2 = V4
8060 V3 = S(NS,2,I):V4 = S(NS,3,I)
8070  GOSUB 4000: REM * VECTEUR *
8080  NEXT I
8090  RETURN : REM ----------------------------------------------
8100  REM
8110  REM
```

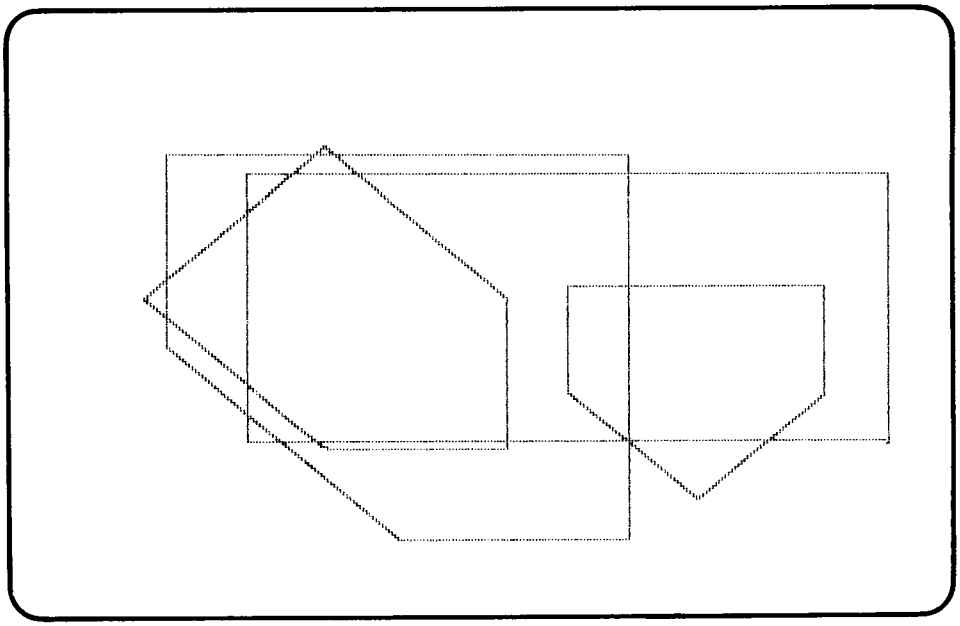
```
9000  REM ------------------------------------------------------------
9002  REM COPIE D'UNE SURFACE
9004  REM ------------------------------------------------------------
9010  FOR I = 1 TO NP
9020 S(NS,1,I) = S(0,1,I)
9030 S(NS,2,I) =  INT (S(0,2,I) + .5)
9040 S(NS,3,I) =  INT (S(0,3,I) + .5)
9050  NEXT I
9060  RETURN : REM ------------------------------------------------------
9070  REM
9080  REM
10000  REM ------------------------------------------------------------
10002  REM ELIMINATION DES SURFACES CACHEES
10004  REM ------------------------------------------------------------
10010  GOSUB 10200: REM * ORDONNEES MIN ET MAX *
10020  FOR YI = YN TO YX
10030  GOSUB 10400: REM * INTERSECTIONS *
10040  GOSUB 10700: REM * TRI LISTE     *
10050  GOSUB 10900: REM * AFFICHER      *
10060  NEXT YI
10070  RETURN : REM ------------------------------------------------------
10080  REM
10090  REM
10200  REM ------------------------------------------------------------
10202  REM ORDONNEES MIN ET MAX
10204  REM ------------------------------------------------------------
10210 YN = 999:YX =  - 999
10220  FOR I = 1 TO NTS
10230  FOR J = 1 TO S(I,0,0)
10240  IF S(I,3,J) < YN THEN YN = S(I,3,J)
10250  IF S(I,3,J) > YX THEN YX = S(I,3,J)
10260  NEXT J
10270  NEXT I
10280  RETURN : REM ------------------------------------------------------
10290  REM
10300  REM
```

```
10400  REM ------------------------------------------------------------
10402  REM INTERSECTIONS
10404  REM ------------------------------------------------------------
10410 K = 0
10420  FOR I = 1 TO NTS
10430  FOR J = 1 TO S(I,0,0) - 1
10440  IF S(I,3,J) > YI AND S(I,3,J + 1) > YI THEN 10640

10450  IF S(I,3,J) < YI AND S(I,3,J + 1) < YI THEN 10640

10460 F1 = S(I,2,J) - S(I,2,J + 1)
10470 F2 = S(I,3,J) - S(I,3,J + 1)
10480 F3 = S(I,2,J):F4 = S(I,3,J) - YI
10490 F5 = 280
10500 DE = F5 * F2
10510  IF DE = 0 THEN 10590
10520 TX = (F3 * F2 - F1 * F4) / DE
10530 TY = (F5 * F4) / DE
10540  IF TX < 0 OR TX > 1 THEN 10640
10550  IF TY < 0 OR TY > 1 THEN 10640
10560 K = K + 1:P(0,K) = 0:P(1,K) = I
10570 P(2,K) =  INT (F5 * TX + .5):P(3,K) = S(I,3,J) - S
    (I,3,J + 1)
10580  GOTO 10640
10590  IF YI <  > S(I,3,J) THEN 10640
10600 K = K + 1:P(0,K) = 1:P(1,K) = I
10610 P(2,K) = S(I,2,J):P(3,K) = YI
10620 K = K + 1:P(0,K) = 1:P(1,K) = I
10630 P(2,K) = S(I,2,J + 1):P(3,K) = YI
10640  NEXT J
10650  NEXT I
10660  RETURN : REM ------------------------------------------------------
10670  REM
10680  REM
```

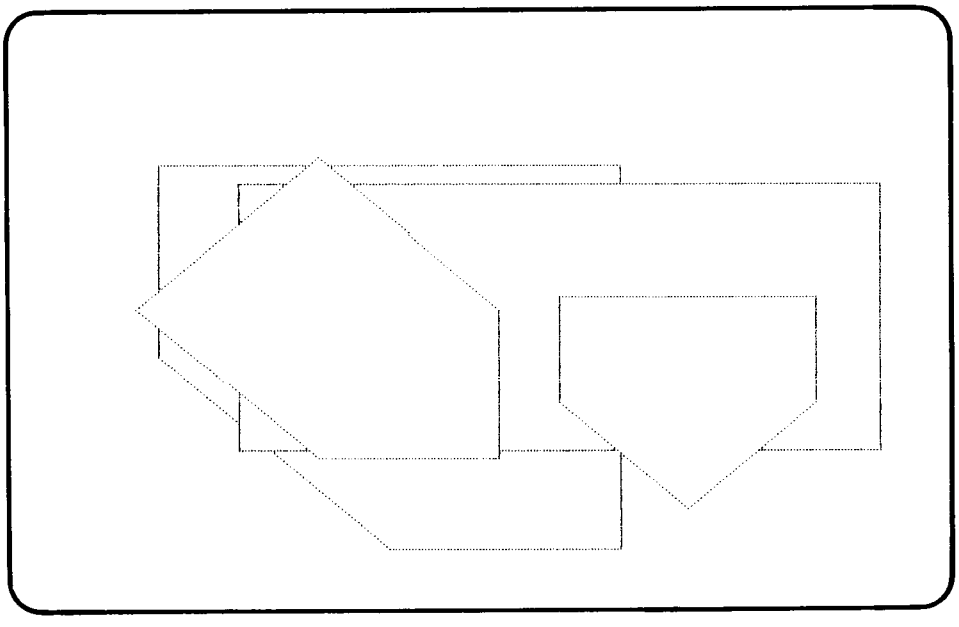
```
10700  REM ------------------------------------------------------------
10702  REM TRI LISTE
10704  REM ------------------------------------------------------------
10710 I = 1
10720  IF P(2,I) < P(2,I + 1) THEN 10780
10730  IF P(2,I) = P(2,I + 1) AND P(0,I) <  = P(0,I + 1)
     THEN 10780
10740  FOR J = 0 TO 3
10750 Z = P(J,I):P(J,I) = P(J,I + 1):P(J,I + 1) = Z
10760  NEXT J
10770 F = 1
10780  IF I < K - 1 THEN I = I + 1: GOTO 10720
10790  IF F <  > 0 THEN F = 0: GOTO 10710
10800 I = 0
10810  FOR J = 1 TO K
10820 I = I + 1:P(0,I) = P(0,J):P(1,I) = P(1,J):P(2,I) =
    P(2,J)
10830  IF P(1,J) <  > P(1,J + 1) OR P(2,J) <  > P(2,J +
    1) THEN 10870
10840  IF P(0,J) <  > P(0,J + 1) THEN I = I - 1: GOTO 10
    870
10850  IF P(3,J) > 0 AND P(3,J + 1) > 0 THEN I = I - 1: GOTO
    10870
10860  IF P(3,J) < 0 AND P(3,J + 1) < 0 THEN I = I - 1
10870  NEXT J
10880 K = I + 1
10890  RETURN : REM ----------------------------------------------
10892  REM
10894  REM
10900  REM ------------------------------------------------------------
10902  REM AFFICHER
10904  REM ------------------------------------------------------------
10910 I = 1:J = 1:L = 0:XI = P(2,I)
10920 J = J + 1
10930  IF J > K THEN 11160
10940  IF P(1,I) >  = P(1,J) THEN 10970
10945  REM  S(PI)<S(PJ)
10950 M = J: GOSUB 11500: REM * PUSH *
10960  GOTO 10920
10970  IF P(1,I) > P(1,J) THEN 11070
10975  REM  S(PI)=S(PJ)
```

```
10980  IF P(0,I) = 0 THEN 11030
10990 V1 = XI:V2 = YI:V3 = P(2,J):V4 = YI: GOSUB 4000
11000 XI = P(2,J)
11010  IF L < > 0 THEN I = PI(L):L = L - 1: GOTO 10920
11020 I = J + 1:J = J + 1:XI = P(2,I): GOTO 10920
11030 X = XI:Y = YI: GOSUB 3000:X = P(2,J): GOSUB 3000
11040 XI = P(2,J)
11050  IF L < > 0 THEN I = PI(L):L = L - 1: GOTO 10920
11060 I = J + 1:J = J + 1:XI = P(2,I): GOTO 10920
11065  REM  S(PI)>S(PJ)
11070  IF P(0,I) = 0 THEN 11120
11080 V1 = XI:V2 = YI:V3 = P(2,J):V4 = YI: GOSUB 4000
11090 M = I: GOSUB 11500: REM * PUSH *
11100 I = J:XI = P(2,J)
11110  GOTO 10920
11115  REM  M(PI)=0
11120 X = XI:Y = YI: GOSUB 3000
11130 M = I: GOSUB 11500: REM  * PUSH *
11140 I = J:XI = P(2,J)
11150  GOTO 10920
11160  RETURN : REM ----------
11170  REM
11180  REM
11500  REM ----------
11502  REM  PUSH
11504  REM ----------
11510 JJ = 1:F = 0
11520  IF L = 0 THEN 11580
11530  FOR II = 1 TO L
11540  IF P(1,PI(II)) = P(1,M) AND P(0,PI(II)) = P(0,M) THEN
    F = 1: GOTO 11560
11550 PI(JJ) = PI(II):JJ = JJ + 1
11560  NEXT II
11570  IF F = 1 THEN F = 0:L = L - 1: GOTO 11590
11580 L = L + 1:PI(L) = M
11590  IF L = 1 THEN 11650
11600 II = 1:F = 0
11610  IF P(1,PI(II)) > = P(1,PI(II + 1)) THEN 11630
11620 Z = PI(II):PI(II) = PI(II + 1):PI(II + 1) = Z:F =
    1
11630  IF II < L - 1 THEN II = II + 1: GOTO 11610
11640  IF F < > 0 THEN 11600
11650  RETURN : REM ----------
11660  REM
11670  REM
```

```
12000  REM ------------------------------------------------------------
12002  REM DEFINITION DES SURFACES
12004  REM ------------------------------------------------------------
12010  FOR I = 1 TO NTS
12020  READ S(I,0,0)
12030  FOR J = 1 TO S(I,0,0)
12040  FOR K = 1 TO 3
12050  READ S(I,K,J)
12060  NEXT K
12070  NEXT J
12080  NEXT I
12085  REM PREMIERE SURFACE
12090  DATA  6
12100  DATA  0,0,0
12110  DATA  1,0,2
12120  DATA  1,2,2
12130  DATA  1,3,1
12140  DATA  1,2,0
12150  DATA  1,0,0
12155  REM DEUXIEME SURFACE
12160  DATA  6
12170  DATA  0,0,0
12180  DATA  1,0,2
12190  DATA  1,1,2
12200  DATA  1,2,1
12210  DATA  1,1,0
12220  DATA  1,0,0
12225  REM TROISIEME SURFACE
12230  DATA  5
12240  DATA  0,0,0
12250  DATA  1,0,2
12260  DATA  1,4,2
12270  DATA  1,4,0
12280  DATA  1,0,0
12285  REM QUATRIEME SURFACE
12290  DATA  6
12300  DATA  0,0,0
12310  DATA  1,0,1
12320  DATA  1,1,2
12330  DATA  1,2,2
12340  DATA  1,2,0
12350  DATA  1,0,0
12360  RETURN : REM ---------------------------------------------------
12370  REM
```



Sans élimination des parties cachées



Avec élimination des parties cachées

Courbes planes

BEZIER

La définition de surfaces par des contours polygonaux s'avère très simple à mettre en oeuvre, nous l'avons vu. Cependant, si le contour à modéliser n'est plus polygonal mais courbe, l'approximation à l'aide de segments peut nécessiter un très grand nombre de points. Les manipulations des surfaces deviennent alors très longues et fastidieuses. Pour résoudre ce problème, le Docteur es-sciences Pierre Bezier a proposé une méthode appliquée dans son système UNISURF, à la Régie Renault, depuis 1972. De cette méthode, nous n'allons voir que la définition des courbes planes de Bezier. Il faut néanmoins savoir qu'elle a été largement développée et que plusieurs voitures ont été dessinées grâce à elle.

HYPOTHESE

Nous allons "oublier" tout ce que nous avons vu jusqu'à présent. Les courbes de Bezier en effet, n'ont que les noms des transformations de commun avec ce que nous avons envisagé, mais n'ont pas toujours les mêmes modalités d'application. Nous allons définir des courbes de Bezier en deux dimensions et pouvoir rapidement juger de l'efficacité de cette étonnante et passionnante modélisation.

MODELE

Pour définir une courbe de Bezier, on définit en premier lieu un polygone ; étrange similitude avec ce que nous avons vu dans les chapitres précédents, n'est-ce-pas ?...

Ce polygone est défini, disons, par (N+1) points Pi. Ces derniers jouent un rôle de première importance car ils servent à la définition mathématique de la courbe de Bezier suivante :

La courbe est définie par l'ensemble discret des points P(t) tels que :

$$P(t) = \sum_{i=0}^{n} Pi\ Bi,n\ (t)$$

où

$$Bi,n\ (t) = C_n^i\ t^i\ (1-t)^{n-i}$$

avec

$$C_n^i = \frac{n!}{i!(n-i)!}\ \text{(coefficients binomiaux)}$$

et

$$Pi = \begin{pmatrix} Xi \\ Yi \end{pmatrix} \qquad P(t) = \begin{pmatrix} X(t) \\ Y(t) \end{pmatrix}$$

En faisant varier t entre les valeurs 0 et 1 d'un incrément faible, on calcule avec cette modélisation mathématique, les points de la courbe définie par le polygone de Bezier.

Pour tracer une courbe, le programme de l'exemple d'exécution saisit les points d'un polygone par appel au sous-programme ***SAISIE DES PARAMETRES***. Ces points servent de base de calcul au sous-programme ***CALCUL DE LA COURBE***, qui utilise CALCUL DES B (I,NP-1,T) pour évaluer les coefficients à appliquer aux points du polygone pour déterminer les coordonnées des points de la courbe. Cette dernière est ensuite tracée par appel au sous-programme ***VECTEUR*** dont l'utilisation permet d'obtenir un tracé plus rapide mais de précision moindre que celle qui aurait été obtenue par le sous-programme ***POINT*** et d'un incrément de valeur inférieure. Le choix s'est porté sur la rapidité dans le programme. Enfin, le polygone de Bezier est tracé sur l'écran.

EXEMPLE D'EXECUTION

Saisie

```
                    COURBE PLANE DE BEZIER


          NOMBRE DE POINTS DU POLYGONE ? 7

            COORDONNEES DU POINT 1 ? 0,0

            COORDONNEES DU POINT 2 ? 46,191

            COORDONNEES DU POINT 3 ? 93,0

            COORDONNEES DU POINT 4 ? 139,191

            COORDONNEES DU POINT 5 ? 186,0

            COORDONNEES DU POINT 6 ? 232,191

            COORDONNEES DU POINT 7 ? 232,191
```

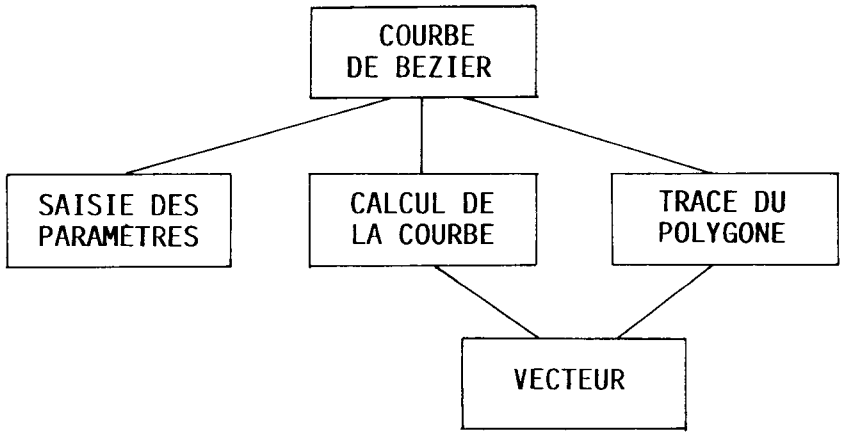
Résultat

Le résultat d'exécution se trouve en haut de la page 203, page au bas de laquelle on trouve un exemple de tracé de plusieurs courbes de polygones différents. Ces exemples montrent l'influence des modifications des coordonnées des points.

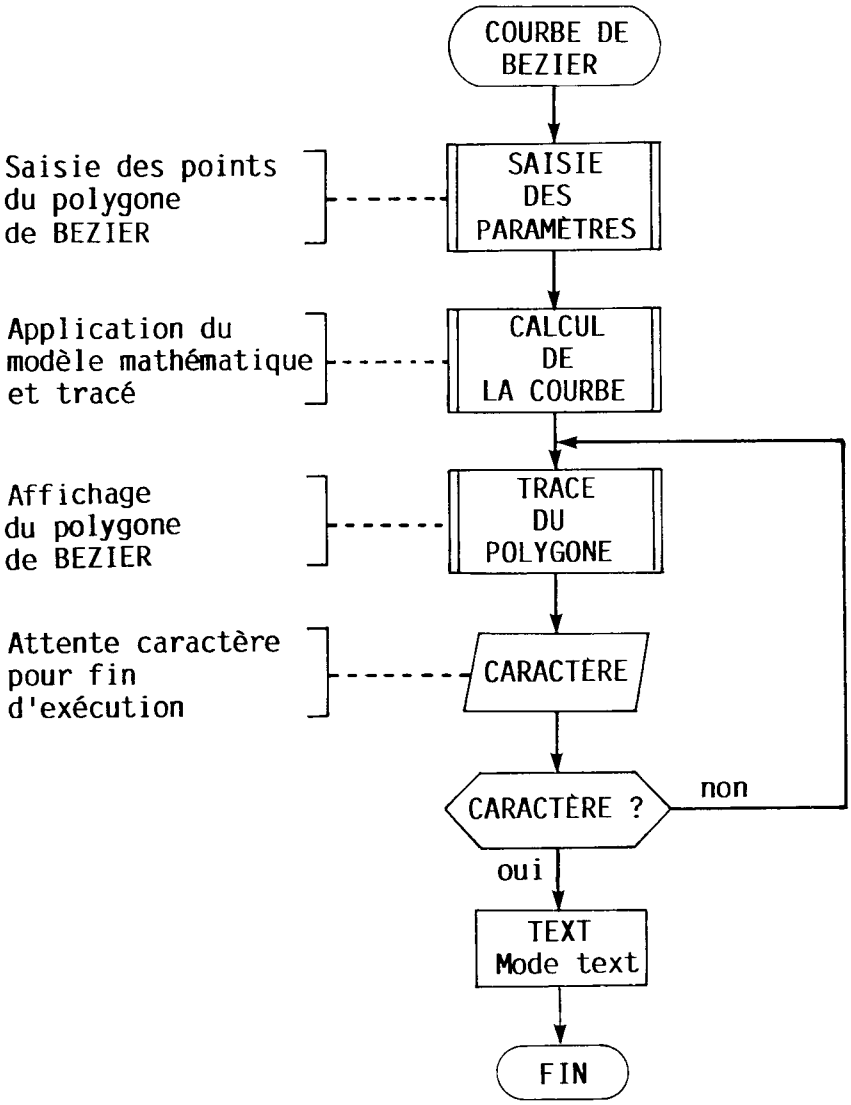
LE PROGRAMME

Traitement

Structure



Organigramme



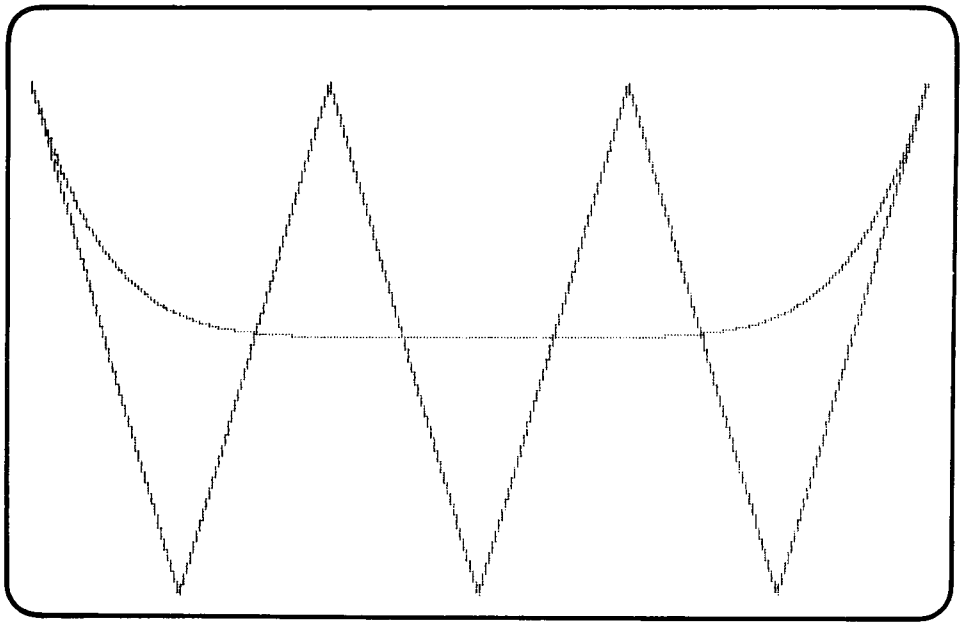
Variables

COURBE DE BEZIER	:	Programme principal
R$	:	Caractère de fin d'exécution
VECTEUR	:	Tracé de vecteurs
V1,V2	:	Coordonnées du premier point du vecteur à tracer
V3,V4	:	Coordonnées du second point du vecteur à tracer
SAISIE DES PARAMETRES	:	Saisie du polygone de BEZIER
NP	:	Nombre de points du polygone
P (2,NP)	:	Tableau des points du polygone
I	:	Indice de saisie des points

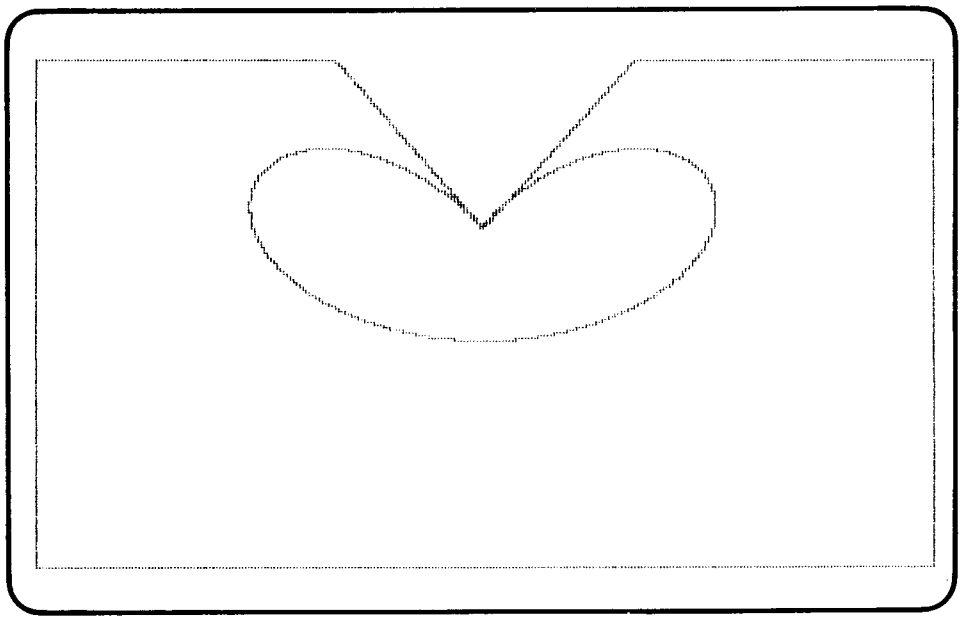
CALCUL DE LA COURBE	:	Tracé de la courbe de Bezier
T	:	Variable de parcours de la courbe
X,Y	:	Coordonnées des points de la courbe
I	:	Indice de parcours des points du polygone
B	:	Coefficient de Bezier
P	:	Tableau des points du polygone
V1,V2,V3,V4	:	Coordonnées transférées du sous-programme VECTEUR
CALCUL DES B (I,NP-1,T)		
T1,T2	:	Variables utilisées pour le calcul des coefficients binomiaux
T	:	Variable de parcours de la courbe
II	:	Indice de boucle
TRACE DU POLYGONE		
I	:	Indice de parcours des points du polygone
V1,V2,V3,V4	:	Variables transférées au sous-programme VECTEUR
NP	:	Nombre de points du polygone

```
§LIST(1000-

 1000  REM ------------------------------------------------
 1002  REM COURBE DE BEZIER
 1004  REM ------------------------------------------------
 1010  GOSUB 4000: REM * SAISIE DES PARAMETRES *
 1020  GOSUB 5000: REM * CALCUL DE LA COURBE   *
 1030  GOSUB 7000: REM * TRACE DU POLYGONE     *
 1040  GET R$: IF R$ = "" THEN 1040
 1050  TEXT
 1060  END : REM ------------------------------------------
 1070  REM
 1080  REM
 3000  REM ------------------------------------------------
 3002  REM VECTEUR
 3004  REM ------------------------------------------------
 3006  REM V3,V4: COORDONNEES DU POINT 2
 3010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 3
    040
 3020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 3
    040
 3030  HPLOT V1,V2 TO V3,V4
 3040  RETURN : REM ----------------------------------------
 3050  REM
 3060  REM
 4000  REM ------------------------------------------------
 4002  REM SAISIE DES PARAMETRES
 4004  REM ------------------------------------------------
 4010  HOME
 4020  PRINT  TAB( 9);"COURBE PLANE DE BEZIER"
 4030  PRINT : PRINT : PRINT
 4040  INPUT "  NOMBRE DE POINTS DU POLYGONE ? ";NP
 4050  DIM P(2,NP)
 4060  PRINT
 4070  FOR I = 0 TO NP - 1
 4080  PRINT "    COORDONNEES DU POINT ";I + 1;
 4090  INPUT " ? ";P(1,I),P(2,I)
 4100  PRINT
 4110  NEXT I
 4120  HOME
 4130  HGR : POKE  - 16302,0: HCOLOR= 3
 4140  RETURN : REM ---------------------------------------
 4150  REM
 4160  REM
```

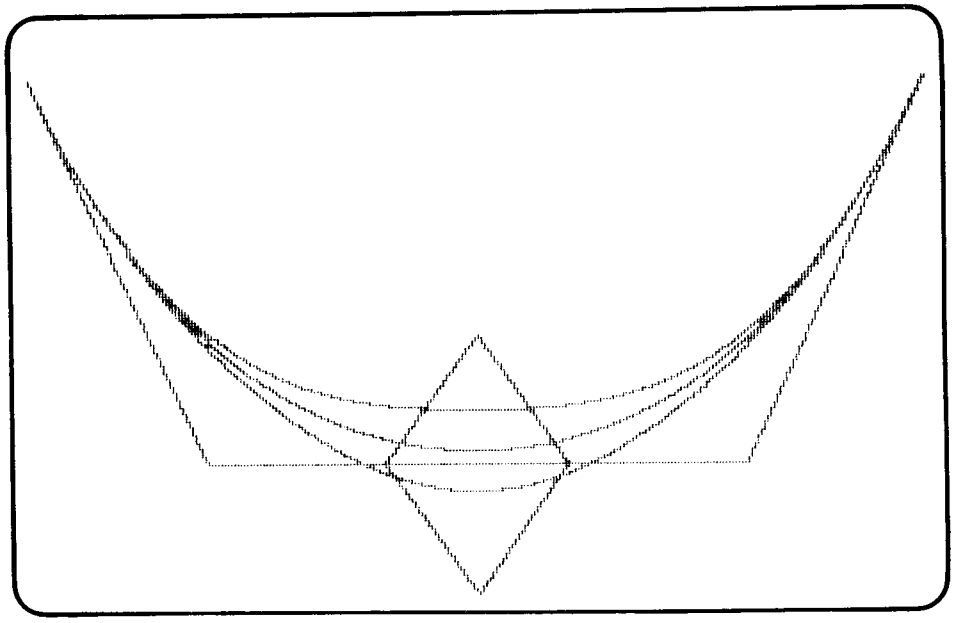
```
5000  REM ------------------------------------------------------
5002  REM CALCUL DE LA COURBE
5004  REM ------------------------------------------------------
5010  FOR T = 0 TO 1 STEP .001
5020 X = 0:Y = 0
5030  FOR I = 0 TO NP - 1
5040  GOSUB 6000: REM * CALCUL DE B(I,NP-1,T)
5050 X = X + P(1,I) * B
5060 Y = Y + P(2,I) * B
5070  NEXT I
5080  IF T = 0 THEN V1 = X:V2 = Y: GOTO 5120
5090 V3 = X:V4 = Y
5100  GOSUB 3000: REM  * VECTEUR *
5110 V1 = V3:V2 = V4
5120  NEXT T
5130  RETURN : REM ---------------------------------------------
5140  REM
5150  REM
6000  REM ------------------------------------------------------
6002  REM CALCUL DES B(I,NP-1,T)
6004  REM ------------------------------------------------------
6010 T1 = 1:T2 = 1
6020  FOR II = NP - I TO NP - 1
6030 T1 = T1 * II
6040  NEXT II
6050  FOR II = 1 TO I
6060 T2 = T2 * II
6070  NEXT II
6080  IF I = 0 THEN T1 = 1:T2 = 1
6090 B = (T1 / T2) * (T ^ I) * ((1 - T) ^ (NP - 1 - I))
6100  RETURN : REM ---------------------------------------------
6110  REM
6120  REM
7000  REM ------------------------------------------------------
7002  REM TRACE DU POLYGONE
7004  REM ------------------------------------------------------
7010  FOR I = 0 TO NP - 2
7020 V1 = P(1,I):V2 = P(2,I):V3 = P(1,I + 1):V4 = P(2,I +
    1)
7030  GOSUB 3000: REM * VECTEUR *
7040  NEXT I
7050  RETURN : REM ---------------------------------------------
7060  REM
```



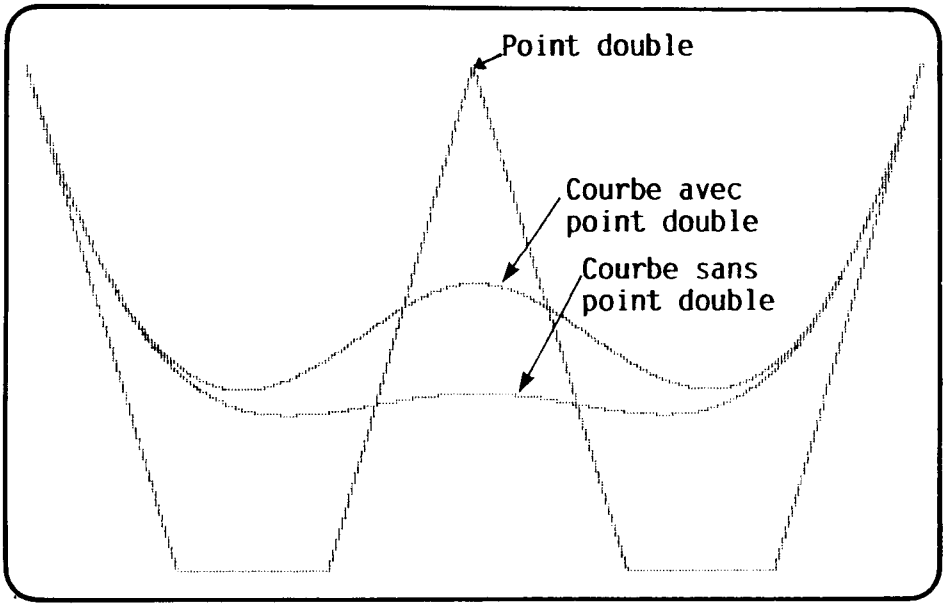
Exemple de tracé de courbe



Exemple de courbe fermée



Exemple de courbe obtenue avec changement de point



Exemple de courbe obtenue avec un point double

B-SPLINE

Dans ce chapitre nous allons envisager une nouvelle modélisation des courbes planes : le modèle d'interpolation B-SPLINE. S'appuyant, comme pour BEZIER, sur un polygone de définition, le calcul des courbes dans ce modèle offre des avantages considérables. En particulier, le changement des coordonnées de l'un des points du polygone ne modifie l'allure d'une courbe que localement et, le nombre de vecteurs du polygone ne fixe plus le degré de la courbe. L'utilisation de la base B-SPLINE revêt ainsi une importance toute particulière que nous ne pouvions laisser sous silence.

HYPOTHESE

Rompu à l'utilisation des courbes de BEZIER, nous n'allons avoir aucune peine à mettre en oeuvre le modèle d'interpolation B-SPLINE des courbes planes dont BEZIER est un cas particulier. En effet, nous allons conserver les mêmes notions que celles développées dans le chapitre précédent et seulement appréhender un concept que nous avions laissé sous silence jusqu'à présent, celui de l'ordre d'une courbe.

MODELE

Dans le chapitre précédent, nous avions défini un polygone de (n+1) points Pi. Conservons cette définition. La courbe au sens B-SPLINE est alors définie par l'ensemble discret des points P(t) tels que :

$$P(t) = \sum_{i=\emptyset}^{n} Pi\ Ni,k\ (t)$$

où

$$Ni,1(t) = \begin{cases} 1 \text{ si } Xi \leqslant t \leqslant Xi+1 \\ 0 \text{ sinon} \end{cases}$$

et

$$Ni,k(t) = \frac{(t-Xi)\ Ni,k-1(t)}{X_{i+k-1}-X_i} + \frac{(Xi+1-t)\ Ni+1,k-1(t)}{X_{i+k}-X_{i+1}}$$

(Convention : 0/0 = 0)

définissent les coefficients B-SPLINE de la ième courbe d'ordre **k** pour lesquels les **Xi** influent sur la flexibilité et sont définis comme suit :

$$X_i = \begin{cases} 0 & \text{si } i < n \\ i-k+1 & \text{si } k \leqslant i \leqslant n \\ n-k+1 & \text{si } i > n \end{cases}$$

Après avoir calculé les **Xi** et les **Ni,1**, pour calculer les points de la courbe, on fait varier **t** entre deux ***Xi*** consécutifs, cela pour tous les **Xi** et l'on est en mesure de tracer les points associés au polygone de définition. Cependant, pour un même polygone, on peut obtenir des résultats différents en faisant varier la variable **k**, ordre de la courbe. Ainsi, pour la limite inférieure k=2, la courbe tracée sera quasiment la même que celle résultant du tracé du polygone. Au contraire pour la limite supérieure k=n+1, la courbe sera celle qui, par interpolation, a les plus faibles pentes. Notons que lorsque l'ordre de la courbe est égal au nombre de points du polygone moins un, on se trouve dans le cas particulier de la courbe de Bezier.

Le programme de l'exemple d'exécution permet l'introduction d'un polygone et de l'ordre de la courbe à tracer par appel du sous-programme ***SAISIE DES PARAMETRES.*** Ce dernier étant exécuté, c'est ***CALCUL DE LA COURBE*** qui s'exécute en appelant ***CALCUL DES XI*** et, ensuite, en calculant les points de la courbe qui sont tracés par le sous-programme ***POINT.*** Après la courbe B-SPLINE c'est, pour terminer, le polygone de définition qui est tracé par le sous-programme ***TRACE DU POLYGONE.***

EXEMPLE D'EXECUTION

Saisie

```
                    COURBE PLANE B-SPLINE


          ORDRE DE LA COURBE ? 6


          NOMBRE DE POINTS DU POLYGONE ? 7

            COORDONNEES DU POINT 1 ? 0,0

            COORDONNEES DU POINT 2 ? 46,191

            COORDONNEES DU POINT 3 ? 93,0

            COORDONNEES DU POINT 4 ? 139,191

            COORDONNEES DU POINT 5 ? 186,0

            COORDONNEES DU POINT 6 ? 232,191

            COORDONNEES DU POINT 7 ? 279,0
```

Résultat

Le résultat d'exécution pour le polygone défini ci-dessus, se trouve en haut de la page 212.

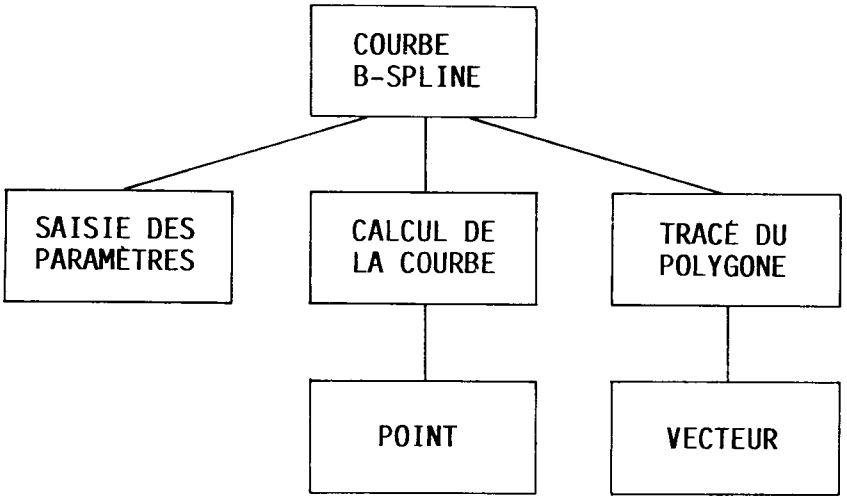
Les autres tracés que l'on peut voir pages suivantes tentent de mettre en valeur les particularités de comportement du modèle B-SPLINE. Il est intéressant de les comparer aux tracés obtenus par la modélisation de BEZIER.

LE PROGRAMME

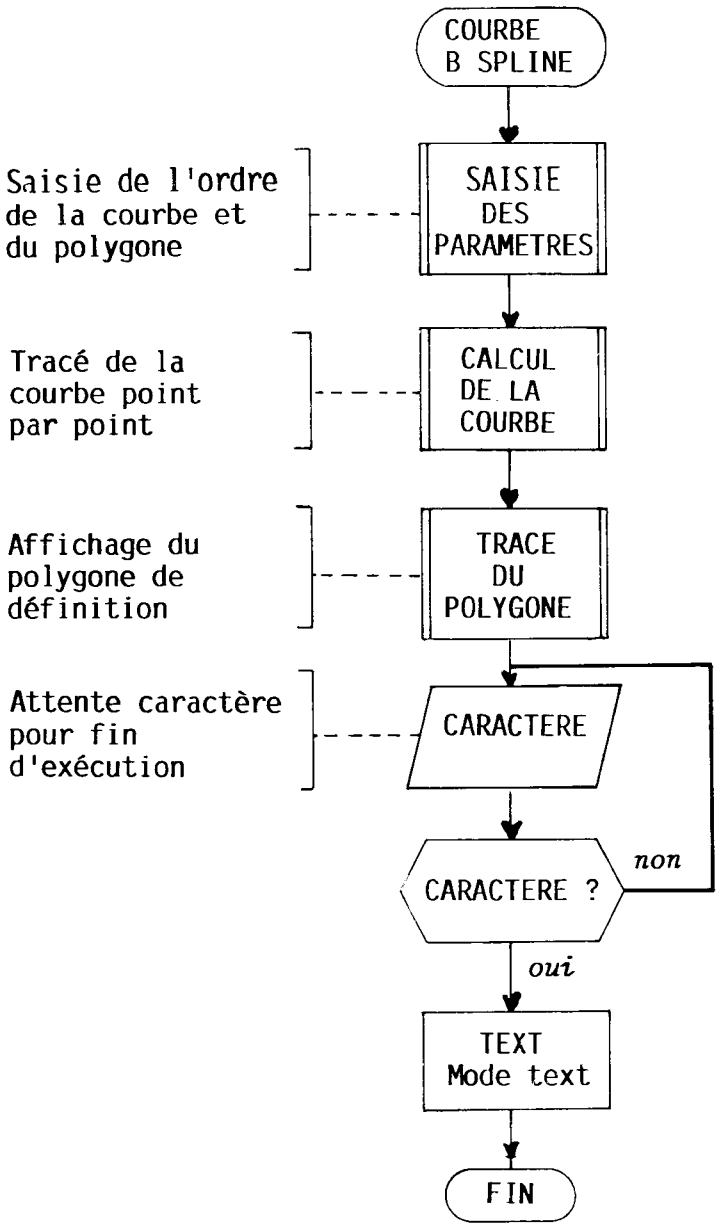
Traitement

Ordre de la courbe
Polygone de définition → COURBE B-SPLINE → *Tracé d'une courbe B-SPLINE et du polygone associé*

Structure



Organigramme



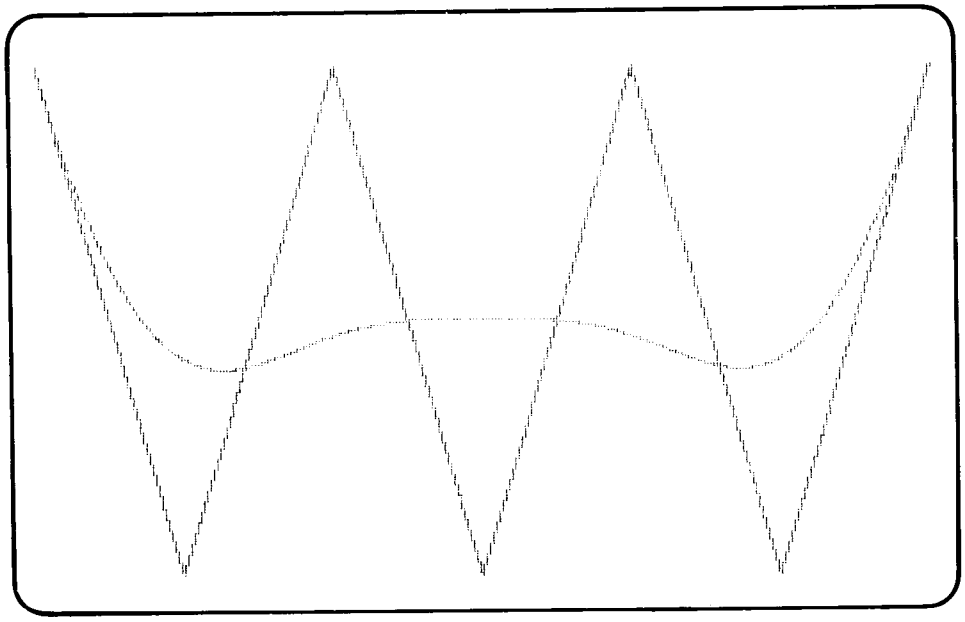
Variables

COURBE B-SPLINE	:	Programme principal
R$	:	Caractère de fin d'exécution
POINT	:	Affichage d'un point
X	:	Abscisse du point à afficher
Y	:	Ordonnée du point à afficher
VECTEUR	:	Affichage d'un vecteur
V1,V2	:	Coordonnées du premier point du vecteur
V3,V4	:	Coordonnées du second point du vecteur
SAISIE DES PARAMETRES	:	Saisie du polygone de définition
O	:	Ordre de la courbe
NP	:	Nombre de points du polygone
P (2,NP-1)	:	Tableau des points du polygone
N (NP+O+1,NP+O+1)	:	Tableau des coefficients B-SPLINE
X (NP+O+1)	:	Tableau des Xi
I	:	Indice de lecture des points du polygone
CALCUL DE LA COURBE	:	Calcul et tracé des points de la courbe B-SPLINE
I,J,K,T	:	Indices de calcul
O	:	Ordre de la courbe
NP	:	Nombre de points du polygone de définition
X (NP+O+1)	:	Tableau des Xi
N (NP+O+1,NP+O+1)	:	Tableau des coefficients B-SPLINE
PAS	:	Définit l'incrément entre les points calculés de la courbe
T1,T2	:	Facteurs des Ni,k
X,Y	:	Coordonnées du point calculé
P (2,NP-1)	:	Tableau des points du polygone
CALCUL DES XI	:	Calcul des Xi tenant compte des points doubles
I	:	Indice de boucle
O	:	Ordre de la boucle
NP	:	Nombre de points du polygone
P (2,NP-1)	:	Tableau des points du polygone
X (NP+O+1)	:	Tableau des Xi calculés
TRACE DU POLYGONE	:	Affichage du polygone de définition
I	:	Indice de parcours des points du polygone
V1...V4	:	Coordonnées transférées au sous-programme VECTEUR
P (2,NP-1)	:	Tableau des points du polygone

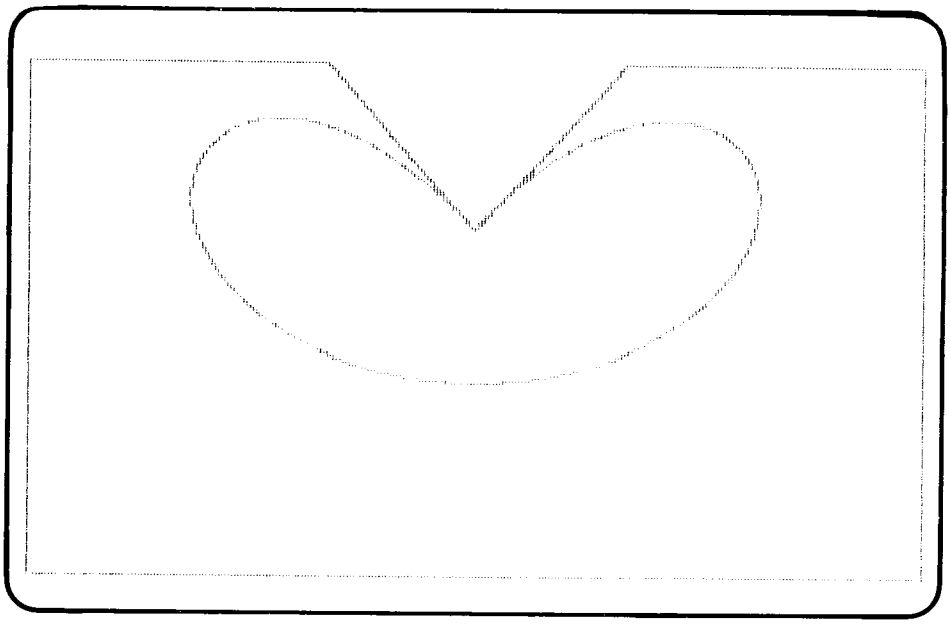
```
]LIST1000-

1000  REM ------------------------------------------------------------
1002  REM COURBE B-SPLINE
1004  REM ------------------------------------------------------------
1010  GOSUB 4000: REM * SAISIE DES PARAMETRES *
1020  GOSUB 5000: REM * CALCUL DE LA COURBE   *
1030  GOSUB 7000: REM * TRACE DU POLYGONE     *
1040  GET R$: IF R$ = "" THEN 1040
1050  TEXT
1060  END : REM ------------------------------------------------------------
1070  REM
1080  REM
2000  REM ------------------------------------------------------------
2002  REM POINT
2004  REM ------------------------------------------------------------
2010  REM TEST DE VALIDITE
2020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
2030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
2040  REM AFFICHAGE DU POINT
2050  HPLOT X,Y
2060  RETURN : REM ------------------------------------------------------------
2070  REM
2080  REM
3000  REM ------------------------------------------------------------
3002  REM VECTEUR
3004  REM ------------------------------------------------------------
3005  REM V1,V2: COORDONNEES DU POINT 1
3006  REM V3,V4: COORDONNEES DU POINT 2
3010  IF V1 < 0 OR V3 < 0 OR V1 > 279 OR V3 > 279 THEN 4
    040
3020  IF V2 < 0 OR V4 < 0 OR V2 > 191 OR V4 > 191 THEN 4
    040
3030  HPLOT V1,V2 TO V3,V4
3040  RETURN : REM ------------------------------------------------------------
3050  REM
3060  REM
```

```
4000  REM ----------
4002  REM SAISIE DES PARAMETRES
4004  REM ----------
4010  HOME
4020  PRINT  TAB( 9);"COURBE PLANE B-SPLINE"
4030  PRINT : PRINT : PRINT
4040  INPUT "  ORDRE DE LA COURBE ? ";O
4050  PRINT : PRINT
4060  INPUT "  NOMBRE DE POINTS DU POLYGONE ? ";NP
4070  DIM P(2,NP - 1),N(NP + O + 1,NP + O + 1),X(NP + O +
    1)
4080  PRINT
4090  FOR I = O TO NP - 1
4100  PRINT "    COORDONNEES DU POINT ";I + 1;
4110  INPUT " ? ";P(1,I),P(2,I)
4120  PRINT
4130  NEXT I
4140  HOME
4150  HGR : POKE  - 16302,O: HCOLOR= 3
4160  RETURN : REM ----------
4170  REM
4180  REM
5000  REM ----------
5002  REM CALCUL DE LA COURBE
5004  REM ----------
5010  GOSUB 6000: REM * CALCUL DES XI *
5020  FOR J = O - 1 TO NP + 1
5030  FOR I = O TO NP + O - 2
5040  IF I <  > J THEN 5060
5050  IF X(I) <  > X(I + 1) THEN N(I,1) = 1: GOTO 5070
5060 N(I,1) = O
5070  NEXT I
5080 PAS = X(NP + O - 1) / 1000
5090  FOR T = X(J) TO X(J + 1) STEP PAS
5100 X = O:Y = O
5110  FOR K = 2 TO O
5120  FOR I = O TO NP - 1
5130  IF N(I,K - 1) = O THEN T1 = O: GOTO 5150
5140 T1 = ((T - X(I)) * N(I,K - 1)) / (X(I + K - 1) - X(
    I))
5150  IF N(I + 1,K - 1) = O THEN T2 = O: GOTO 5170
5160 T2 = ((X(I + K) - T) * N(I + 1,K - 1)) / (X(I + K) -
    X(I + 1))
5170 N(I,K) = T1 + T2
5180  IF K <  > O THEN 5210
5190 X = X + P(1,I) * N(I,K)
5200 Y = Y + P(2,I) * N(I,K)
5210  NEXT I
5220  NEXT K
5230  IF X = O AND Y = O THEN X = P(1,NP - 1):Y = P(2,NP
    - 1)
5240  GOSUB 2000: REM * POINT *
5250  NEXT T
5260  NEXT J
5270  RETURN : REM ----------
5280  REM
5290  REM
```

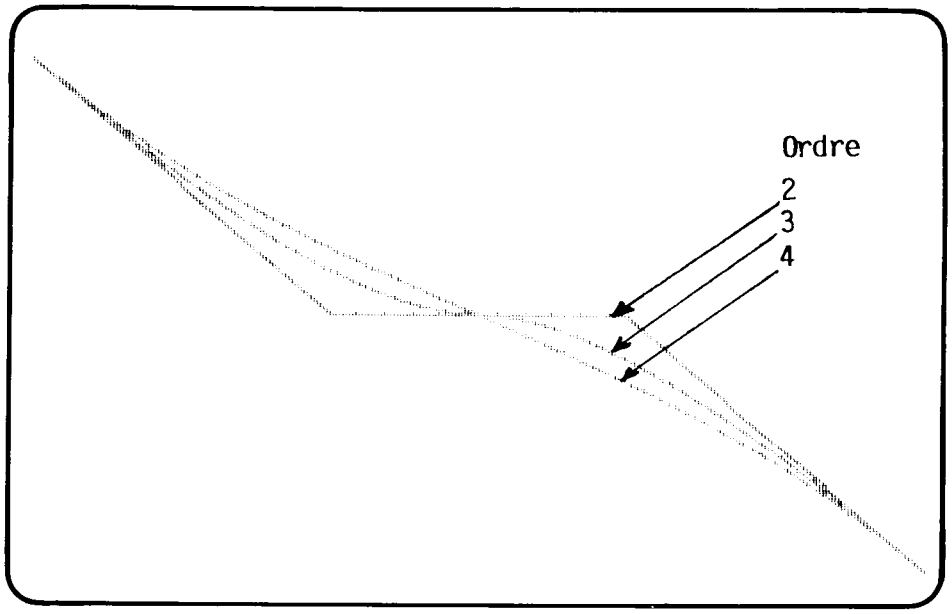
```
6000  REM ------------------------------------------------------------
6002  REM CALCUL DES XI
6004  REM ------------------------------------------------------------
6010  FOR I = 0 TO NP + 0 - 1
6020  IF I > 0 - 1 THEN 6050
6030 X(I) = 0
6040  GOTO 6120
6050  IF I < NP + 1 THEN 6070
6060  GOTO 6090
6070  IF P(1,I - 0) <  > P(1,I - 0 + 1) THEN 6110
6080  IF P(2,I - 0) <  > P(2,I - 0 + 1) THEN 6110
6090 X(I) = X(I - 1)
6100  GOTO 6120
6110 X(I) = X(I - 1) + 1
6120  NEXT I
6130  RETURN : REM ------------------------------------------------------
6140  REM
6150  REM
7000  REM ------------------------------------------------------------
7002  REM TRACE DU POLYGONE
7004  REM ------------------------------------------------------------
7010  FOR I = 0 TO NP - 2
7020 V1 = P(1,I):V2 = P(2,I):V3 = P(1,I + 1):V4 = P(2,I +
     1)
7030  GOSUB 3000: REM * VECTEUR *
7040  NEXT I
7050  RETURN : REM ------------------------------------------------------
7060  REM
```



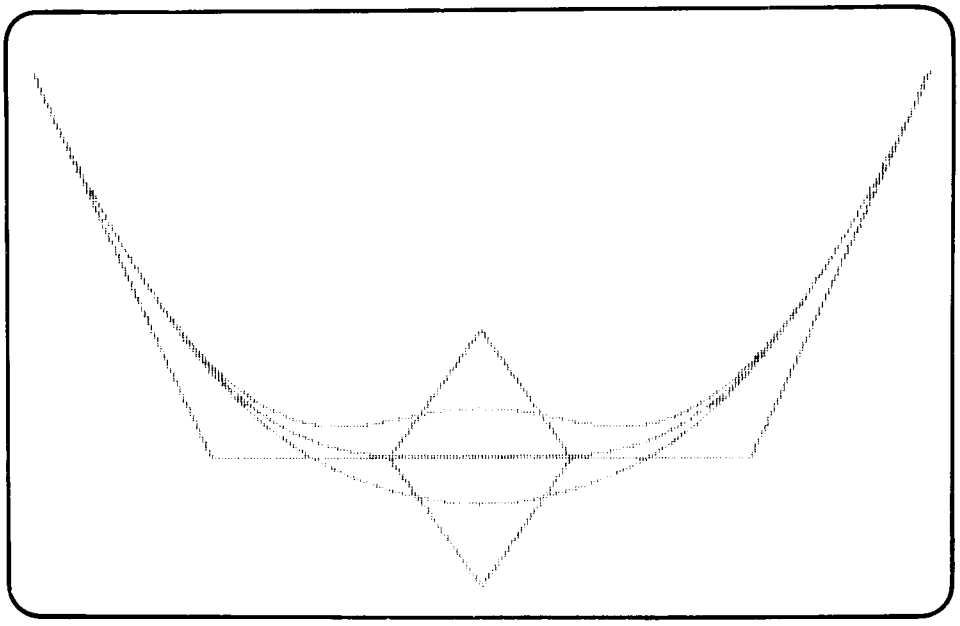
Exemple de courbe



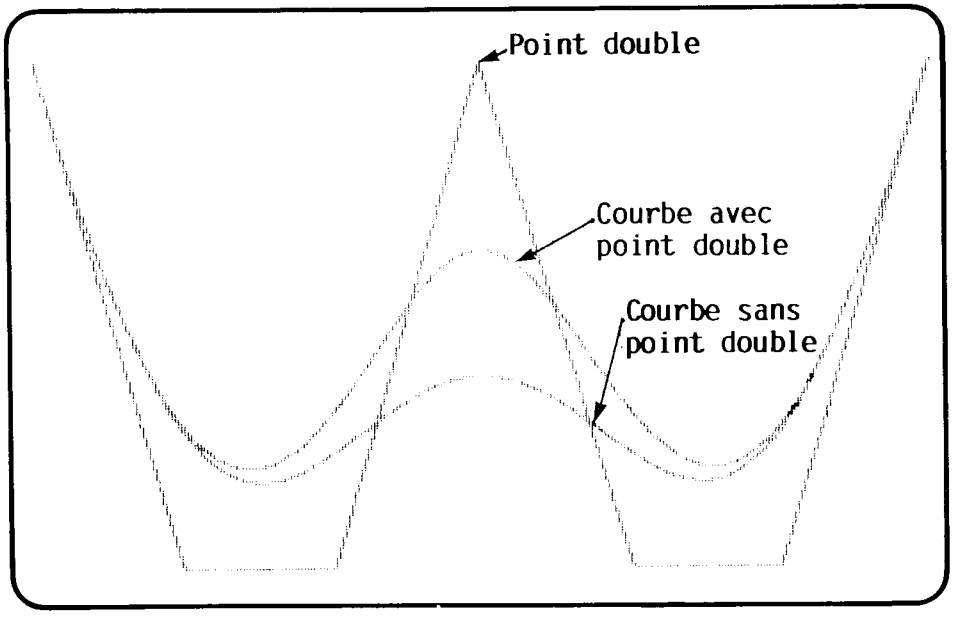
Exemple de courbe fermée



Tous les ordres d'une même courbe



Exemple de courbe obtenue avec le changement d'un point



Exemple de courbe avec un point double

Quatrième partie

EXPRESSION GRAPHIQUE INTERACTIVE

TRACÉS INTERACTIFS

Dans ce chapitre nous allons mettre au point un programme permettant à l'utilisateur d'exprimer directement ses volontés graphiques sur l'écran, sans passer par l'intermédiaire d'équations et de paramétrages.

L'utilisateur pourra, en tapant une touche, agir directement sur l'affichage des points en déterminant la direction et le sens du déplacement de ce que l'on appelerait en langage LOGO : la tortue.

HYPOTHESE

L'intervention de l'utilisateur étant déterministe, relativement à l'exécution du programme, l'interface homme-machine utilisée revêt une importance particulière. L'action, nous l'avons dit, est effectuée par l'intermédiaire d'une touche, et donc, par l'intermédiaire du clavier. Or, le marché actuel des micro-ordinateurs propose trois types principaux de claviers, proposant les mêmes facilités (touches fonctions mises à part), mais dont la disposition des touches diffère.

Si nous considérons les claviers des PET/CBM, par exemple, nous voyons qu'il est divisé en deux parties. La première permet l'accès à tous les caractères alpha et aux caractères spéciaux, alors que la seconde, placée sur la droite, permet la saisie des caractères numériques et des signes algébriques. C'est cette seconde partie que nous utiliserons dans ce chapitre afin de commander la tortue car, nous le verrons, sa disposition convient parfaitement.

Par contre, si nous regardons l'APPLE, son clavier est constitué d'un seul bloc qui propose l'ensemble des caractères ASCII sous la disposition QWERTY. Alors, afin d'obtenir un sous-ensemble de touches suffisamment bien disposées pour commander efficacement la tortue, nous choisirons d'utiliser les touches N, M, , J, L, U, I, O à la place des touches 1, 2, 3, 4, 5, 6, 7, 8 et 9, respectivement.

Le programme permet l'utilisation des deux claviers que nous avons mentionnés plus haut, mais ne prévoit pas l'utilisation du troisième type de clavier qui, en fait, est une autre version du clavier de l'APPLE, la version AZERTY. Pour utiliser un tel clavier, les modifications à apporter au programme sont extrêmement simples. Il suffit, dans la routine ***INTERACTIF*** de repérer les caractères utilisés pour un clavier QWERTY et d'en changer les valeurs par le sous-ensemble des caractères que vous aurez choisi pour commander la tortue puisque tel est notre but.

MODELE

Le programme proposé dans ce chapitre est adapté à l'APPLE. La version du clavier choisi est QWERTY et les touches de commande ont la disposition suivante :

U	I	O
J	K	L
N	M	,

L'affectation choisie, des fonctions aux touches, est alors celle qui suit :

Touche	Direction du tracé
N	Vers le bas à gauche en diagonale
M	Vers le bas verticalement
,	Vers le bas à droite en diagonale
J	Vers la gauche horizontalement
K	Effacement de l'écran
L	Vers la droite horizontalement
U	Vers le haut à gauche en diagonale
I	Vers le haut verticalement
O	Vers le haut à droite en diagonale

Notons aussi l'utilisation de la touche "F" du clavier qui permet de mettre fin à l'exécution du programme.

Lors du lancement du programme, le point de départ du dessin se trouve en position 'HOME' ou encore en coordonnées : (Ø,Ø)

mais, il suffit, pour commencer un dessin à partir d'une autre origine, de déplacer le point de départ par les touches de commandes, puis d'effacer l'écran par la touche "K".

Le déplacement du point qui permet le tracé d'une figure est obtenu en considérant que l'écran est muni d'un repère dont l'un des axes (X) est horizontal et l'autre (Y) est vertical et, appuyer sur une touche de commande, permet d'incrémenter les coordonnées de la position du point courant. En répétant ce processus, il est clair que nous arrivons à tracer une ligne continue comme nous le souhaitions.

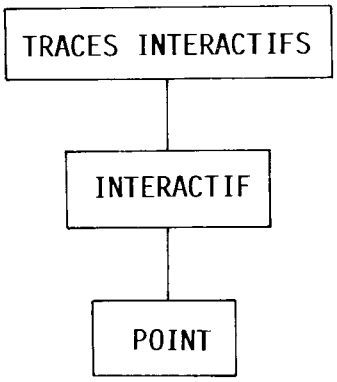
EXEMPLE D'UTILISATION

Essayez, comme nous l'avons fait, de tracer la figure de la page 222. Vous le constatez, ce n'est pas si simple.

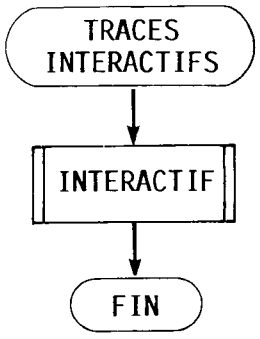
LE PROGRAMME

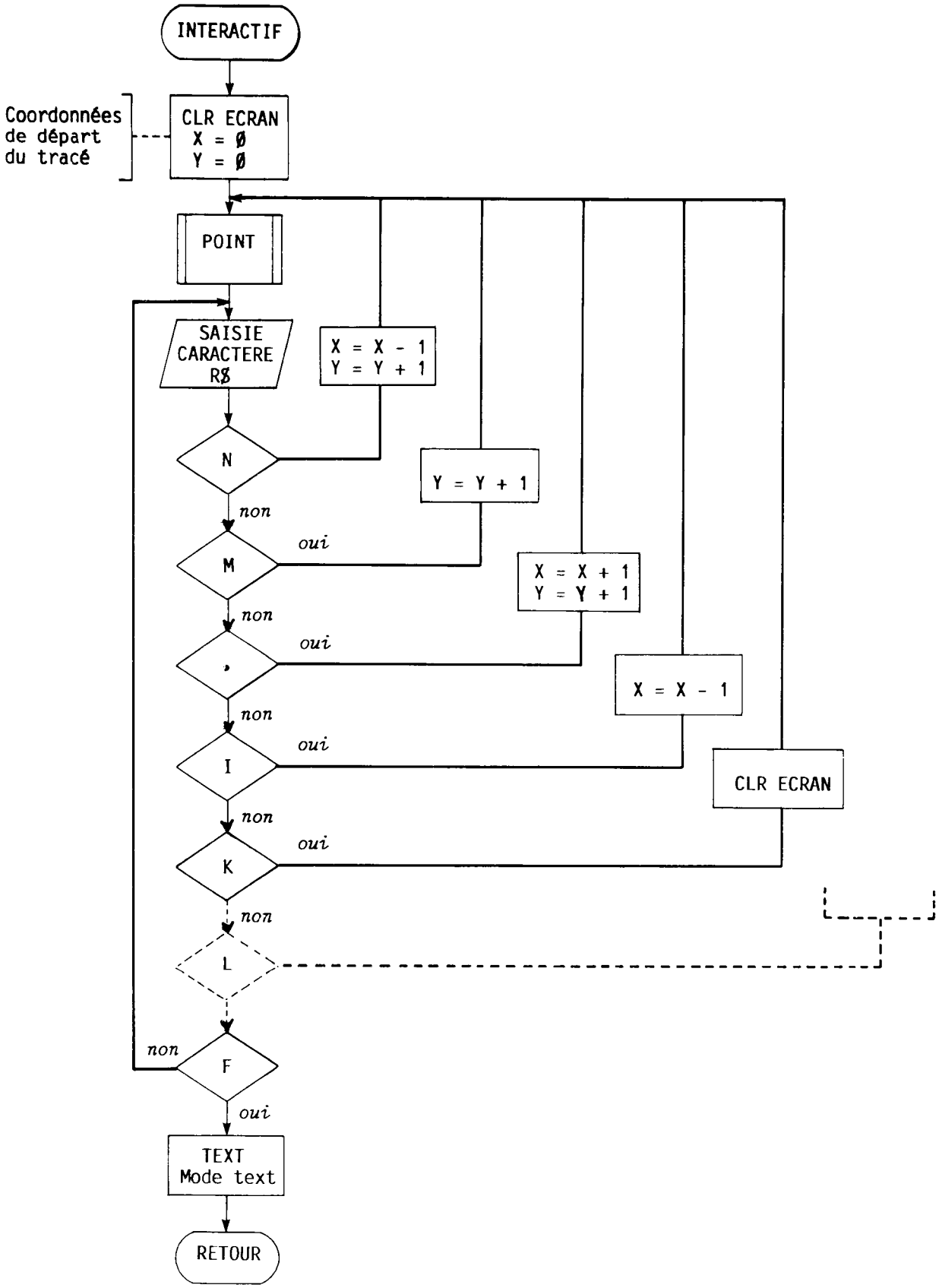
Traitement

Structure



Organigrammes




INTERACTIF
Coordonnées de départ du tracé
CLR ECRAN
X = Ø
Y = Ø
POINT
SAISIE CARACTERE RØ
X = X - 1
Y = Y + 1
N
Y = Y + 1
non
M
oui
X = X + 1
Y = Y + 1
non
oui
X = X - 1
non
I
oui
CLR ECRAN
non
K
oui
non
L
non
F
oui
TEXT
Mode text
RETOUR

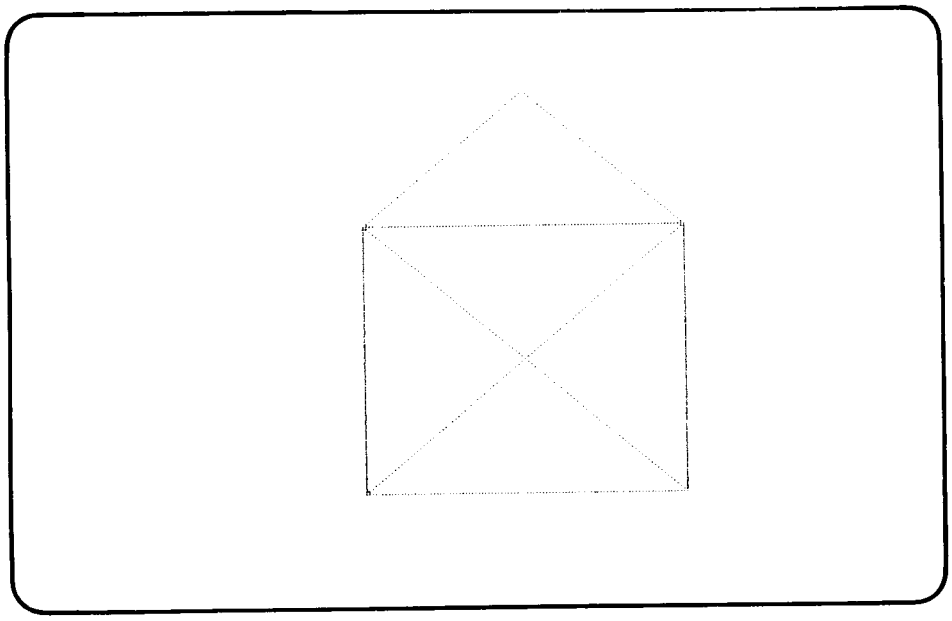
Variables

POINT	:	Affichage d'un point
X		Abscisse du point à afficher
Y		Ordonnée du point à afficher
INTERACTIF	:	Affichage de points en mode interactif
X,Y		Coordonnées calculées du point à afficher
R$		Variable contenant le caractère saisi au clavier et permettant de diriger le tracé interactif

Particularité

Telle que se présente la routine ***INTERACTIF***, il est très simple d'implanter de nouvelles fonctions, commandées elles aussi par la saisie d'un caractère. Une fonction intéressante pourrait être, par exemple, la recopie d'écran sur imprimante qui permettrait de conserver une trace de vos futurs chefs-d'oeuvres.

```
§LIST1000-

1000  REM ------------------------------------------------------------
1002  REM TRACES INTERACTIFS
1004  REM ------------------------------------------------------------
1010  REM
1020  GOSUB 4000: REM * INTERACTIF *
1030  REM
1040  END : REM ------------------------------------------------------
1050  REM
1060  REM
3000  REM ------------------------------------------------------------
3002  REM POINT
3004  REM ------------------------------------------------------------
3010  REM TEST DE VALIDITE
3020  IF X < 0 OR  INT (X + .5) > 279 THEN 3060
3030  IF Y < 0 OR  INT (Y + .5) > 191 THEN 3060
3040  REM AFFICHAGE DU POINT
3050  HPLOT X,Y
3060  RETURN : REM ---------------------------------------------------
3070  REM
3080  REM
4000  REM ------------------------------------------------------------
4002  REM INTERACTIF
4004  REM ------------------------------------------------------------
4005  REM PAGE 1 GRAPHIQUE
4010  HGR : POKE  - 16302,0: HCOLOR= 3
4015  REM (X,Y)=(0,0) : POSITION DE DEPART
4020 X = 0:Y = 0
4025  REM AFFICHAGE POINT
4030  GOSUB 3000: REM * POINT *
4035  REM SAISIE D'UN CARACTERE
4040  GET R$
4045  REM N => DIRECTION SUD-OUEST
4050  IF R$ = "N" THEN X = X - 1:Y = Y + 1
4055  REM M => DIRECTION SUD
4060  IF R$ = "M" THEN Y = Y + 1
4065  REM , => DIRECTION SUD-EST
4070  IF R$ = "," THEN X = X + 1:Y = Y + 1
4075  REM J => DIRECTION OUEST
4080  IF R$ = "J" THEN X = X - 1
4085  REM K => EFACEMENT ECRAN
4090  IF R$ = "K" THEN  HOME : HGR : POKE  - 16302,0
4095  REM L => DIRECTION EST
4100  IF R$ = "L" THEN X = X + 1
4105  REM U => DIRECTION NORD-OUEST
4110  IF R$ = "U" THEN X = X - 1:Y = Y - 1
4115  REM I => DIRECTION NORD
4120  IF R$ = "I" THEN Y = Y - 1
4125  REM O => DIRECTION NORD-EST
4130  IF R$ = "O" THEN X = X + 1:Y = Y - 1
4135  REM F => FIN D'EXECUTION
4140  IF R$ <  > "F" THEN 4030
4150  HOME : TEXT
4160  RETURN : REM ---------------------------------------------------
4170  REM
```



Exemple de tracé interactif

Annexe

X compris entre 0 et 79 et Y compris entre 0 et 50. A cette occasion, si vous n'avez pas fait "CLEAR ECRAN", c'est-à-dire, si vous n'avez pas effacé totalement le contenu de l'écran, avant de faire vos essais, vous pourrez constater qu'il est possible de mélanger : texte et caractères graphiques.

LE PROGRAMME

Lorsque l'exécution est lancée (SYS826), le programme va chercher, en mémoire écran, le contenu du caractère pointé par les coordonnées X,Y du point qu'il s'agit d'allumer. En fonction de ce contenu, il détermine ensuite le caractère à afficher pour que les points déjà allumés ne soient pas éteints lors de l'allumage de ce nouveau point. Il cherche donc un complément au caractère déjà affiché sur l'écran. Il ne reste plus qu'à transmettre le nouveau caractère vers la mémoire écran, ce qui se traduit au niveau machine par le chargement de l'adresse mémoire, pointée par les coordonnées du point à allumer, par le code du caractère complément.

Particularité

Bien que l'on puisse considérer ***GESTION GRAPHIQUE DE L'ECRAN*** comme un programme à part entière, il est présenté ici comme un sous-programme et pourra, étant donné le numérotage de ses lignes, être intégré à tous les programmes de ce livre.

Cependant, une fois chargé dans le buffer du second magnétophone à cassette, il n'est pas nécessaire de le recharger à chaque fois que l'on utilise un nouveau programme BASIC.

EVOLUTIONS ET PERSPECTIVES FUTURES

Le programme ***GESTION GRAPHIQUE DE L'ECRAN*** constitue, dans sa version actuelle, le minimum à mettre en oeuvre pour afficher l'un quelconque des 4000 points ainsi disponibles sur l'écran. De nombreuses améliorations sont possibles.

Nous pourrons, par exemple, supprimer l'étape qui consiste à passer les arguments X et Y par des instructions POKE en contrôlant ce passage directement au niveau machine. Signalons, pour nous aider à réaliser cette modification, les ouvrages :

- CLEFS POUR PET/CBM
  "Comment ?" Numéro 11 Page 101
  Daniel-Jean DAVID
  Editions du PSI
- LA COMMODE (N° 5)
  Le magazine des ordinateurs Commodore
  "Un petit programme graphique" Page 39
  Jean-Paul LAURENT

Cette dernière publication va d'ailleurs plus loin car, elle propose, malheureusement en basse résolution, mais il nous suffira de l'adapter, trois fonctions :

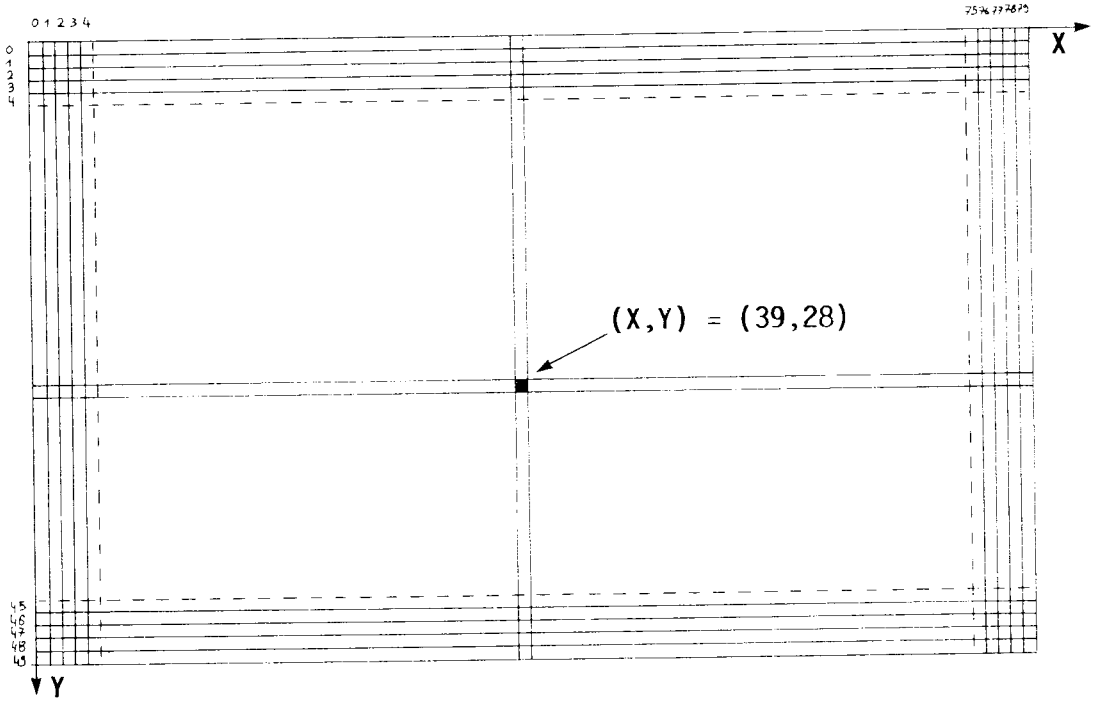
. La fonction **SET** ⟶ **SYS (...), Y, X, 1**
. La fonction **RESET** ⟶ **SYS (...), Y, X, 0**
. La fonction **POINT**⟶**W = 0 : SYS (...), Y, X, Z**

qui permet de connaître la valeur éteint/allumé d'un caractère de l'écran.

Nous pouvons encore améliorer le programme ***GESTION GRAPHIQUE DE L'ECRAN*** en supprimant l'instruction SYS. Pour cela, nous utiliserons l'article de B. MICHEL, auteur des "ROM désassemblées du PET/CBM" (BCM), paru dans la COMMODE N° 5 page 47 qui propose l'ajout d'instructions nouvelles au BASIC standard.

En regroupant ces deux améliorations, nous pourrons sans difficulté, disposer de nouvelles instructions très pratiques d'emploi telles que : **SET (X,Y) ; RESET (X,Y) ; POINT (X,Y,Z)**...etc.

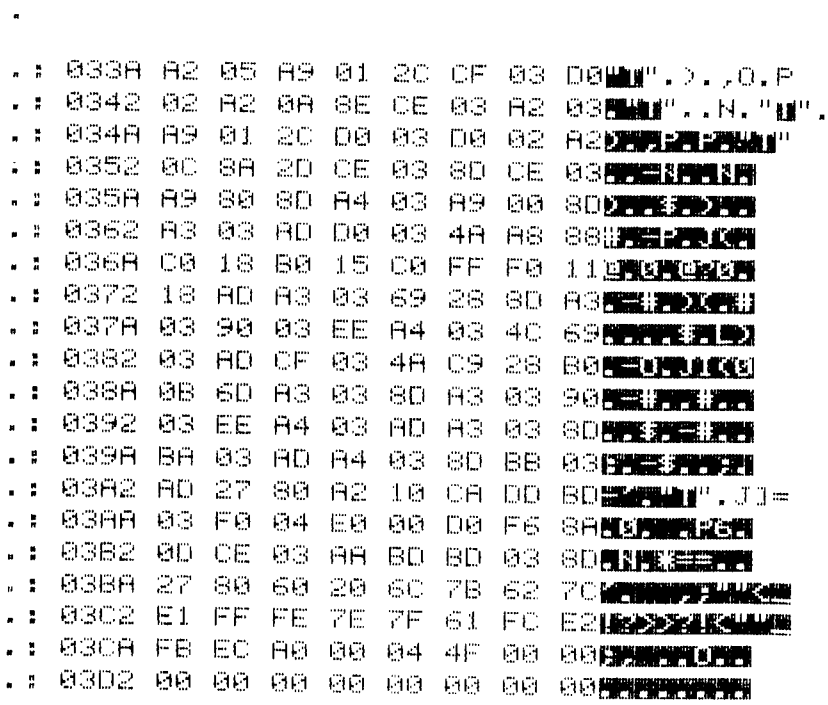
Cette remarque est très importante car, en poussant plus loin le même raisonnement, nous pouvons conclure qu'en quelques chapitres, nous sommes en mesure de concevoir un véritable langage graphique. En effet, les chapitres de ce livre nous permettent de définir (en Basic) les éléments de base du graphisme et il nous suffit de transcrire les routines BASIC en langage machine, de créer de nouvelles instructions pour disposer d'un véritable langage graphique.



Repérage des points sur l'écran

```
2000 REM------------------------------------------------------------
2002 REM*****      GESTION GRAPHIQUE DE L'ECRAN      *****
2004 REM------------------------------------------------------------
2005 :
2006 :REM  COORDONNEES DU POINT :
2007 :REM                                   X EN 975
2008 :REM                                   Y EN 976
2009 :
2010 :FOR I=826 TO 976:READ D:POKEI,D:NEXT I
2020 :DATA 162,5,169,1,44,207,3,208
2030 :DATA 2,162,10,142,206,3,162
2040 :DATA 3,169,1,44,208,3,208,2
2050 :DATA 162,12,138,45,206,3,141
2060 :DATA 206,3,169,128,141,164,3
2070 :DATA 169,0,141,163,3,173,208
2080 :DATA 3,74,168,136,192,24,176
2090 :DATA 21,192,255,240,17,24,173
2100 :DATA 163,3,105,40,141,163,3
2110 :DATA 144,3,238,164,3,76,105
2120 :DATA 3,173,207,3,74,201,40
2130 :DATA 176,11,109,163,3,141,163
2140 :DATA 3,144,3,238,164,3,173
2150 :DATA 163,3,141,186,3,173,164
2160 :DATA 3,141,187,3,173,0,128
2170 :DATA 162,16,202,221,189,3,240
2180 :DATA 4,224,0,208,246,138,13
2190 :DATA 206,3,170,189,189,3,141
2200 :DATA 0,128,96,32,108,123,98
2210 :DATA 124,225,255,254,126,127
2220 :DATA 97,252,226,251,236,160
2230 :DATA 0,1,79,49
2240 RETURN:------------------------------ GESTION ECRAN -----
2250 :
```

Contenu mémoire



Liste désassemblée

```
READY.

B*
     PC   IRQ   SR AC XR YR SP
.; 106F  1355  32 00 10 00 F6
.

., 033A A2 05    LDX #$05
., 033C A9 01    LDA #$01
., 033E 2C CF 03 BIT $03CF
., 0341 D0 02    BNE $0345
., 0343 A2 0A    LDX #$0A
., 0345 8E CE 03 STX $03CE
., 0348 A2 03    LDX #$03
., 034A A9 01    LDA #$01
., 034C 2C D0 03 BIT $03D0
., 034F D0 02    BNE $0353
., 0351 A2 0C    LDX #$0C
., 0353 8A       TXA
., 0354 2D CE 03 AND $03CE
., 0357 8D CE 03 STA $03CE
., 035A A9 80    LDA #$80
., 035C 8D A4 03 STA $03A4
., 035F A9 00    LDA #$00
., 0361 8D A3 03 STA $03A3
., 0364 AD D0 03 LDA $03D0
., 0367 4A       LSR
., 0368 A8       TAY
., 0369 88       DEY
., 036A C0 18    CPY #$18
., 036C B0 15    BCS $0383
., 036E C0 FF    CPY #$FF
., 0370 F0 11    BEQ $0383
., 0372 18       CLC
., 0373 AD A3 03 LDA $03A3
., 0376 69 28    ADC #$28
., 0378 8D A3 03 STA $03A3
., 037B 90 03    BCC $0380
., 037D EE A4 03 INC $03A4
., 0380 4C 69 03 JMP $0369
., 0383 AD CF 03 LDA $03CF
., 0386 4A       LSR
```

```
.; 0387 C9 28     CMP ##$28
.; 0389 B0 0B     BCS $0396
.; 038B 6D A3 03  ADC $03A3
.; 038E 8D A3 03  STA $03A3
.; 0391 90 03     BCC $0396
.; 0393 EE A4 03  INC $03A4
.; 0396 AD A3 03  LDA $03A3
.; 0399 8D BA 03  STA $03BA
.; 039C AD A4 03  LDA $03A4
.; 039F 8D BB 03  STA $03BB
.; 03A2 AD 27 80  LDA $8027
.; 03A5 A2 10     LDX ##$10
.; 03A7 CA        DEX
.; 03A8 DD BD 03  CMP $03BD,X
.; 03AB F0 04     BEQ $03B1
.; 03AD E0 00     CPX #$00
.; 03AF D0 F6     BNE $03A7
.; 03B1 8A        TXA
.; 03B2 0D CE 03  ORA $03CE
.; 03B5 AA        TAX
.; 03B6 BD BD 03  LDA $03BD,X
.; 03B9 8D 27 80  STA $8027
.; 03BC 60        RTS
.; 03BD 20 6C 7B  JSR $7B6C
.; 03C0 62        ???
.; 03C1 7C        ???
.; 03C2 E1 FF     SBC ($FF,X)
.; 03C4 FE 7E 7F  INC $7F7E,X
.; 03C7 61 FC     ADC ($FC,X)
.; 03C9 E2        ???
.; 03CA FB        ???
.; 03CB EC A0 00  CPX $00A0
.; 03CE 04        ???
.; 03CF 4F        ???
.; 03D0 00        BRK
.; 03D1 00        BRK
.; 03D2 00        BRK
.; 03D3 00        BRK
.
```

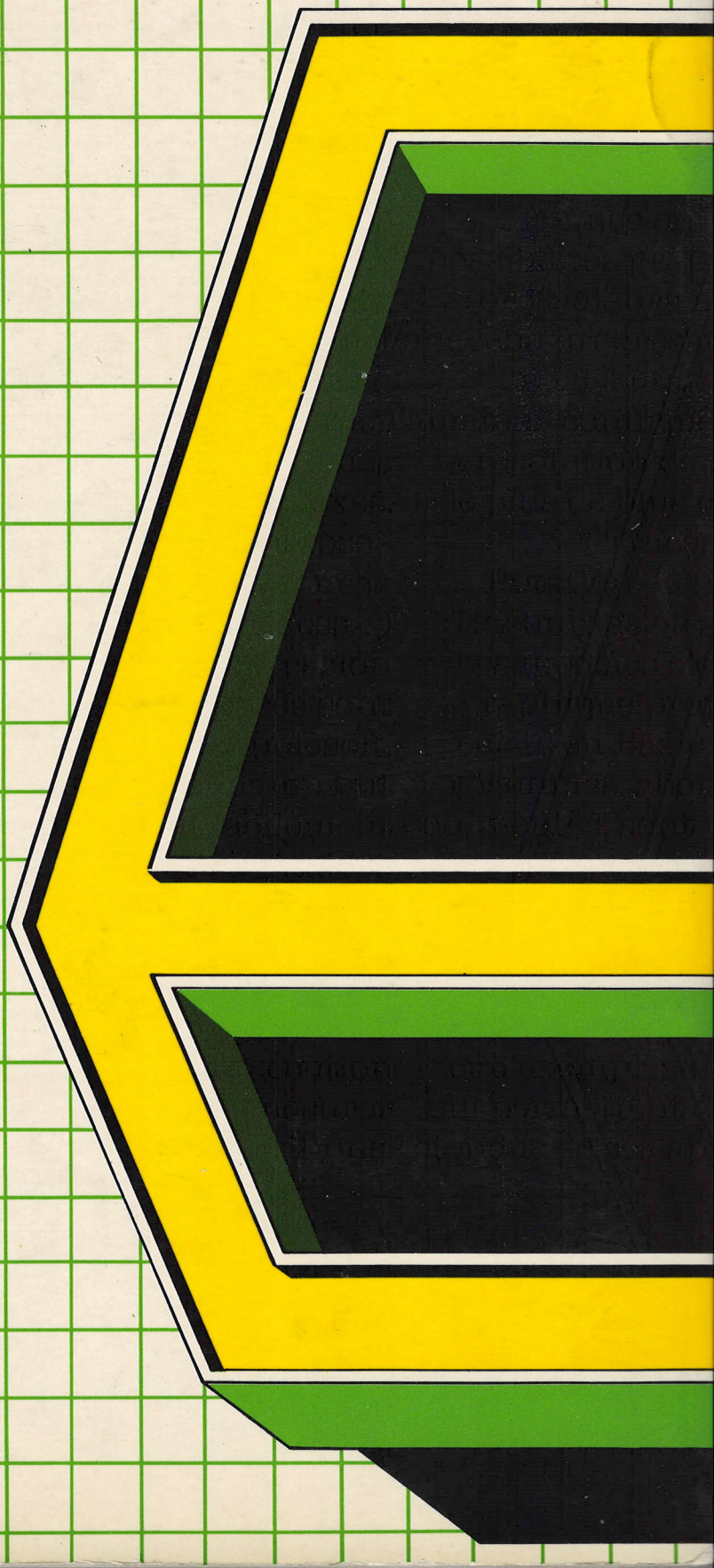
Achevé d'imprimer en février 1984
sur les presses de l'imprimerie Laballery et C^ie
58500 Clamecy
Dépôt légal : janvier 1984

N° d'impression : 401030
N° d'édition : 86595-122-1
ISBN : 2-86595-122-7

130 FF ******

modèles d'expression graphique

Cet ouvrage expose un ensemble de techniques visant la mise en œuvre, des possibilités graphiques des ordinateurs individuels.
Sa démarche progressive permet au débutant comme à l'amateur confirmé d'aborder la résolution de problèmes de plus de plus complexes.
(Tracés d'ellipse - Rotation de polygones - Hachurage de surfaces...)
Les modèles d'expression graphique proposés sont écrits en Basic Applesoft.
Les programmes, abondamment commentés, sont facilement adaptables à d'autres ordinateurs individuels.
Chaque modèle est l'occasion d'une analyse, d'un ou plusieurs programmes et d'exemples d'exécution qui permettent de mesurer l'étendue de ses applications.



EDITIONS DU P.S.I.
BP 86
France - 77402 lagny s/marne cedex
ISBN : 2.86595.122.7
imprimé en France

couverture A GRAPH